

Rec'd PCT/PTO 12 JAN 2005

ST/JP2004/005562

日本国特許庁
JAPAN PATENT OFFICE

19. 4. 2004

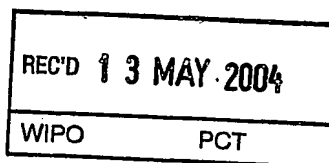
紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: 2003年 5月13日

出願番号
Application Number: 特願2003-133941
[ST. 10/C]: [JP2003-133941]

出願人
Applicant(s): ソニー株式会社

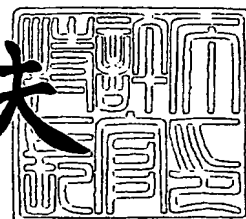


PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH
RULE 17.1(a) OR (b)

2004年 2月19日

特許庁長官
Commissioner,
Japan Patent Office

今井康夫



BEST AVAILABLE COPY

出証番号 出証特2004-3011367

【書類名】 特許願

【整理番号】 0390400801

【提出日】 平成15年 5月13日

【あて先】 特許庁長官殿

【国際特許分類】 H03M 13/03

【発明者】

【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社
内

【氏名】 横川 峰志

【発明者】

【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社
内

【氏名】 宮内 俊之

【発明者】

【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社
内

【氏名】 飯田 康博

【特許出願人】

【識別番号】 000002185

【氏名又は名称】 ソニー株式会社

【代理人】

【識別番号】 100082131

【弁理士】

【氏名又は名称】 稲本 義雄

【電話番号】 03-3369-6479

【手数料の表示】

【予納台帳番号】 032089

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9708842

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 復号装置

【特許請求の範囲】

【請求項 1】 LDPC(Low Density Parity Check)符号の復号装置であって、
P×Pの単位行列、その単位行列のコンポーネントである 1 のうちの 1 個以上が
0 になった行列である準単位行列、前記単位行列もしくは準単位行列をサイクリ
ックシフトした行列であるシフト行列、前記単位行列、準単位行列、もしくはシ
フト行列のうちの複数の和である和行列、またはP×Pの 0 行列を構成行列として
、前記LDPC符号の検査行列が、複数の前記構成行列の組合せで表される場合にお
いて、

前記LDPC符号の復号のためのP個のチェックノードの演算を同時に行うチェッ
クノード計算手段と、

前記LDPC符号の復号のためのP個のバリエブルノードの演算を同時に行うバリ
アブルノード計算手段と

を備えることを特徴とする復号装置。

【請求項 2】 請求項 1 に記載の復号装置であって、

前記チェックノード計算手段は、チェックノードの演算を行うP個のチェック
ノード計算器を有し、

前記バリエブルノード計算手段は、バリエブルノードの演算を行うP個のバリ
アブルノード計算器を有する

ことを特徴とする復号装置。

【請求項 3】 請求項 1 に記載の復号装置であって、

前記P個のチェックノードの演算、または前記P個のバリエブルノードの演算の
結果得られるP個の枝に対応するメッセージデータを同時に読み書きするメッセ
ージ記憶手段をさらに備える

ことを特徴とする復号装置。

【請求項 4】 請求項 3 に記載の復号装置であって、

前記メッセージ記憶手段は、チェックノード演算時に読み出される枝に対応す
るメッセージデータを、検査行列の 1 を行方向に詰めるように格納する

ことを特徴とする復号装置。

【請求項5】 請求項3に記載の復号装置であって、

前記メッセージ記憶手段は、バリエブルノード演算時に読み出される枝に対応するメッセージデータを、検査行列の1を列方向に詰めるように格納することを特徴とする復号装置。

【請求項6】 請求項3に記載の復号装置であって、

前記メッセージ記憶手段は、前記検査行列を表す構成行列のうちの、重みが2以上の構成行列について、その構成行列を、重みが1の単位行列、準単位行列、またはシフト行列の和の形で表現したときの、その重みが1の単位行列、準単位行列、またはシフト行列に属するP個の枝に対応するメッセージを、同一のアドレスに格納する

ことを特徴とする復号装置。

【請求項7】 請求項3に記載の復号装置であって、

前記メッセージ記憶手段は、行数/P個のFIFOと、列数/P個のFIFOとで構成され、

前記行数/P個のFIFOと列数/P個のFIFOは、それぞれ、前記検査行列の行と列の重みに対応するワード数を有する

ことを特徴とする復号装置。

【請求項8】 請求項3に記載の復号装置であって、

前記メッセージ記憶手段は、RAM(Random Access Memory)で構成され、

前記RAMは、前記メッセージデータを、読み出される順番に詰めて格納し、格納位置順に読み出す

ことを特徴とする復号装置。

【請求項9】 請求項1に記載の復号装置であって、

受信情報を格納するとともに、P個の前記受信情報を同時に読み出す受信情報記憶手段をさらに備える

ことを特徴とする復号装置。

【請求項10】 請求項9に記載の復号装置であって、

前記受信情報記憶手段は、前記受信情報を、前記バリエブルノードの演算に必

要となる順番に読み出すことができるように格納することを特徴とする復号装置。

【請求項 11】 請求項 1 に記載の復号装置であって、
前記P個のチェックノードの演算、または前記P個のバリエブルノードの演算の結果得られるメッセージをサイクリックシフトするサイクリックシフト手段をさらに備える

ことを特徴とする復号装置。

【請求項 12】 請求項 11 に記載の復号装置であって、
前記サイクリックシフト手段は、バレルシフタで構成される
ことを特徴とする復号装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、復号装置に関し、特に、低密度パリティ検査符号 (LDPC符号) による符号化が施された符号の復号を行う復号装置に関する。

【0002】

【従来の技術】

近年、例えば、移動体通信や深宇宙通信といった通信分野、及び地上波又は衛星デジタル放送といった放送分野の研究が著しく進められているが、それに伴い、誤り訂正符号化及び復号の効率化を目的として符号理論に関する研究も盛んに行われている。

【0003】

符号性能の理論的限界としては、いわゆるシャノン (C. E. Shannon) の通信路符号化定理によって与えられるシャノン限界が知られている。符号理論に関する研究は、このシャノン限界に近い性能を示す符号を開発することを目的として行われている。近年では、シャノン限界に近い性能を示す符号化方法として、例えば、並列連接畳み込み符号 (PCCC (Parallel Concatenated Convolutional Codes)) や、縦列連接畳み込み符号 (SCCC (Serially Concatenated Convolutional Codes)) といった、いわゆるターボ符号化 (Turbo coding) と呼ばれる手法が開発されて

いる。また、これらのターボ符号が開発される一方で、古くから知られる符号化方法である低密度パリティ検査符号(Low Density Parity Check codes) (以下、LDPC符号という) が脚光を浴びつつある。

【0004】

LDPC符号は、R. G. Gallagerによる「R. G. Gallager, "Low Density Parity Check Codes", Cambridge, Massachusetts: M. I. T. Press, 1963」において最初に提案されたものであり、その後、「D. J. C. MacKay, "Good error correcting codes based on very sparse matrices", Submitted to IEEE Trans. Inf. Theory, IT-45, pp. 399-431, 1999」や、「M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi and D. A. Spielman, "Analysis of low density codes and improved designs using irregular graphs", in Proceedings of ACM Symposium on Theory of Computing, pp. 249-258, 1998」等において再注目されるに至ったものである。

【0005】

LDPC符号は、近年の研究により、ターボ符号等と同様に、符号長を長くしていくにしたがって、シャノン限界に近い性能が得られることがわかりつつある。また、LDPC符号は、最小距離が符号長に比例するという性質があることから、その特徴として、ブロック誤り確率特性がよく、さらに、ターボ符号等の復号特性において観測される、いわゆるエラーフロア現象が殆ど生じないことも利点として挙げられる。

【0006】

以下、このようなLDPC符号について具体的に説明する。なお、LDPC符号は、線形符号であり、必ずしも2元である必要はないが、ここでは、2元であるものとして説明する。

【0007】

LDPC符号は、そのLDPC符号を定義する検査行列(parity check matrix)が疎なものであることを最大の特徴とするものである。ここで、疎な行列とは、行列のコンポーネントの"1"の個数が非常に少なく構成されるものであり、疎な検査行列をHで表すものとする、そのような検査行列としては、例えば、図1に示す

ように、各行のハミング重み ("1"の数) (weight)が"3"であり、且つ、各列のハミング重みが"6"であるもの等がある。

【0008】

このように、各行及び各列のハミング重みが一定である検査行列Hによって定義されるLDPC符号は、レギュラーLDPC符号と称される。一方、各行及び各列のハミング重みが一定でない検査行列Hによって定義されるLDPC符号は、イレギュラーLDPC符号と称される。

【0009】

このようなLDPC符号による符号化は、検査行列Hに基づいて生成行列Gを生成し、この生成行列Gを2元の情報メッセージに対して乗算することによって符号語を生成することで実現される。具体的には、LDPC符号による符号化を行う符号化装置は、まず、検査行列Hの転置行列 H^T との間に、式 $GH^T=0$ が成立する生成行列Gを算出する。ここで、生成行列Gが、 $k \times n$ 行列である場合には、符号化装置は、生成行列Gに対してkビットからなる情報メッセージ(ベクトルu)を乗算し、nビットからなる符号語c ($= uG$)を生成する。この符号化装置によって生成された符号語は、値が"0"の符号ビットが"+1"に、値が"1"の符号ビットが"1"にといったようにマッピングされて送信され、所定の通信路を介して受信側において受信されることになる。

【0010】

一方、LDPC符号の復号は、Gallagerが確率復号(Probabilistic Decoding)と称して提案したアルゴリズムであって、バリアブルノード(variable node (メッセージノード(message node)ともいう。))と、チェックノード(check node)とからなる、いわゆるタナーグラフ(Tanner graph)上での確率伝播(belief propagation)によるメッセージ・パッシング・アルゴリズムによって行うことが可能である。ここで、以下、適宜、バリアブルノードとチェックノードを、単に、ノードともいう。

【0011】

しかしながら、確率復号においては、各ノード間で受け渡されるメッセージが実数値であることから、解析的に解くためには、連続した値をとるメッセージの

確率分布そのものを追跡する必要があり、非常に困難を伴う解析を必要とすることになる。そこで、Gallagerは、LDPC符号の復号アルゴリズムとして、アルゴリズムA又はアルゴリズムBを提案している。

【0012】

LDPC符号の復号は、一般的には、図2に示すような手順にしたがって行われる。なお、ここでは、受信値を $U_0(u_{0i})$ とし、チェックノードから出力されるメッセージを u_j とし、バリアブルノードから出力されるメッセージを v_i とする。また、ここでは、メッセージとは、値の"0"らしさを、いわゆる対数尤度比(log likelihood ratio)で表現した実数値である。

【0013】

まず、LDPC符号の復号においては、図2に示すように、ステップS11において、受信値 $U_0(u_{0i})$ が受信され、メッセージ u_j が"0"に初期化されるとともに、繰り返し処理のカウンタとしての整数をとる変数 k が"0"に初期化され、ステップS12に進む。ステップS12において、受信値 $U_0(u_{0i})$ に基づいて、式(1)に示す演算を行うことによってメッセージ v_i が求められ、さらに、このメッセージ v_i に基づいて、式(2)に示す演算を行うことによってメッセージ u_j が求められる。

【0014】

【数1】

$$v_i = u_{0i} + \sum_{j=1}^{d_v-1} u_j \quad \dots (1)$$

【0015】

【数2】

$$\tanh\left(\frac{u_j}{2}\right) = \prod_{i=1}^{d_c-1} \tanh\left(\frac{v_i}{2}\right) \quad \dots (2)$$

【0016】

ここで、式(1)と式(2)における d_v と d_c は、それぞれ、検査行列Hの縦方

向（列）と横方向（行）の”1”の個数を示す任意に選択可能とされるパラメータであり、例えば、(3,6)符号の場合には、 $d_v=3$, $d_c=6$ となる。

【0017】

なお、式（1）または（2）の演算においては、それぞれ、メッセージを出力しようとする枝(edge)から入力されたメッセージを、和または積演算のパラメータとしては用いないことから、和または積演算の範囲が、1乃至 d_v-1 または1乃至 d_c-1 となっている。また、式（2）に示す演算は、実際には、2入力 v_1 , v_2 に対する1出力で定義される式（3）に示す関数 $R(v_1, v_2)$ のテーブルを予め作成しておき、これを式（4）に示すように連続的（再帰的）に用いることによって行われる。

【0018】

【数3】

$$x = 2 \tanh^{-1} \{ \tanh(v_1/2) \tanh(v_2/2) \} = R(v_1, v_2) \quad \dots (3)$$

【0019】

【数4】

$$u_j = R(v_1, R(v_2, R(v_3, \dots R(v_{d_c-2}, v_{d_c-1}))) \quad \dots (4)$$

【0020】

ステップS12では、さらに、変数 k が”1”だけインクリメントされ、ステップS13に進む。ステップS13では、変数 k が所定の繰り返し復号回数 N よりも大きいかが判定される。ステップS13において、変数 k が N よりも大きくないと判定された場合、ステップS12に戻り、以下、同様の処理が繰り返される。

【0021】

また、ステップS13において、変数 k が N よりも大きいと判定された場合、ステップS14に進み、式（5）に示す演算を行うことによって最終的に出力する復号結果としてのメッセージ v_i が求められて出力され、LDPC符号の復号処理が終了する。

【0022】

【数5】

$$v_i = u_{0i} + \sum_{j=1}^{d_v} u_j \quad \dots (5)$$

【0023】

ここで、式(5)の演算は、式(1)の演算とは異なり、バリエブルノードに接続している全ての枝からの入力メッセージを用いて行われる。

【0024】

このようなLDPC符号の復号は、例えば(3,6)符号の場合には、図3に示すように、各ノード間でメッセージの授受が行われる。なお、図3における“=”で示すノード(バリエブルノード)では、式(1)に示した演算が行われ、“+”で示すノード(チェックノード)では、式(2)に示した演算が行われる。特に、アルゴリズムAにおいては、メッセージを2元化し、“+”で示すノードにて、 d_c-1 個の入力メッセージの排他的論理和演算を行い、“=”で示すノードにて、受信値Rに対して、 d_v-1 個の入力メッセージが全て異なるビット値であった場合には、符号を反転して出力する。

【0025】

また、一方で、近年、LDPC符号の復号の実装法に関する研究も行われている。実装方法について述べる前に、まず、LDPC符号の復号を模式化して説明する。

【0026】

図4は、(3,6)LDPC符号(符号化率1/2、符号長12)の検査行列(parity check matrix)の例である。LDPC符号の検査行列は、図5のように、タナグラフを用いて書き表すことができる。ここで、図5において、“+”で表わされるのが、チェックノードであり、“=”で表わされるのが、バリエブルノードである。チェックノードとバリエブルノードは、それぞれ、検査行列の行と列に対応する。チェックノードとバリエブルノードとの間の結線は、枝(edge)であり、検査行列の“1”に相当する。即ち、検査行列の第j行第i列のコンポーネントが1である場合には、図5において、上からi番目のバリエブルノード(“=”のノード)と、上

から j 番目のチェックノード ("+"のノード) とが、枝により接続される。枝は、バリエブルノードに対応する符号ビットが、チェックノードに対応する拘束条件を持つことを表わす。なお、図 5 は、図 4 の検査行列のタナグرافとなっている。

【0027】

LDPC符号の復号方法であるサムプロダクトアルゴリズム (Sum Product Algorithm) は、バリエブルノードの演算とチェックノードの演算とを繰り返し行う。

【0028】

バリエブルノードでは、図 6 のように、式 (1) の演算を行う。すなわち、図 6 において、計算しようとしている枝に対応するメッセージ v_i は、バリエブルノードに繋がっている残りの枝からのメッセージ u_1 および u_2 と、受信情報 u_{0i} を用いて計算される。他の枝に対応するメッセージも同様に計算される。

【0029】

チェックノードの演算について説明する前に、式 (2) を、式 $a \times b = \exp \{ \ln(|a|) + \ln(|b|) \} \times \text{sign}(a) \times \text{sign}(b)$ の関係を用いて、式 (6) のように書き直す。但し、 $\text{sign}(x)$ は、 $x \geq 0$ のとき 1 であり、 $x < 0$ のとき -1 である。

【0030】

【数 6】

$$\begin{aligned} u_j &= 2 \tanh^{-1} \left(\prod_{i=1}^{d_c-1} \tanh \left(\frac{v_i}{2} \right) \right) \\ &= 2 \tanh^{-1} \left[\exp \left\{ \sum_{i=1}^{d_c-1} \ln \left(\left| \tanh \left(\frac{v_i}{2} \right) \right| \right) \right\} \times \prod_{i=1}^{d_c-1} \text{sign} \left(\tanh \left(\frac{v_i}{2} \right) \right) \right] \\ &= 2 \tanh^{-1} \left[\exp \left\{ - \left(\sum_{i=1}^{d_c-1} - \ln \left(\tanh \left(\frac{|v_i|}{2} \right) \right) \right) \right\} \times \prod_{i=1}^{d_c-1} \text{sign}(v_i) \right] \\ &\quad \dots (6) \end{aligned}$$

【0031】

更に、 $x \geq 0$ において、 $\phi(x) = \ln(\tanh(x/2))$ と定義すると、 $\phi^{-1}(x) = 2 \tanh^{-1}(e^{-x})$ であるから、式 (6) は、式 (7) のように書くことができる。

【0032】

【数7】

$$u_j = \phi^{-1} \left(\sum_{i=1}^{d_c-1} \phi(|v_i|) \right) \times \prod_{i=1}^{d_c-1} \text{sign}(v_i) \quad \dots (7)$$

【0033】

チェックノードでは、図7のように、式(7)の演算を行う。すなわち、図7において、計算しようとしている枝に対応するメッセージ u_j は、チェックノードに繋がっている残りの枝からのメッセージ v_1, v_2, v_3, v_4, v_5 を用いて計算される。他の枝に対応するメッセージも同様に計算される。

【0034】

なお、関数 $\phi(x)$ は、 $\phi(x) = \ln((e^x + 1)/(e^x - 1))$ とも表すことができ、 $x > 0$ において、 $\phi(x) = \phi^{-1}(x)$ である。関数 $\phi(x)$ および $\phi^{-1}(x)$ をハードウェアに実装する際には、LUT(Look Up Table)を用いて実装される場合があるが、両者共に同一のLUTとなる。

【0035】

サムプロダクトアルゴリズムをハードウェアに実装する場合、式(1)で表わされるバリアブルノード演算および式(7)で表わされるチェックノード演算とを、適度な回路規模と動作周波数で繰り返し行うことが必要である。

【0036】

復号装置の実装の例として、まず、単純に各ノードの演算を一つずつ順次行うことによって復号を行う場合(full serial decoding)の実装法について説明する。

【0037】

なお、ここでは、例えば、図8の、30(行)×90(列)の検査行列で表現される符号(符号化率2/3、符号長90)を復号することとする。図8の検査行列の1の数は269であり、従って、そのタナグラフでは、枝の数は269個となる。ここで、図8の検査行列では(後述する図15においても同様)、0を、"."で表現している。

【0038】

図9は、LDPC符号の1回復号を行う復号装置の構成例を示している。

【0039】

図9の復号装置では、その動作する1クロック(clock)ごとに、1つの枝に対応するメッセージが計算される。

【0040】

即ち、図9の復号装置は、1つの受信用メモリ104、2つの枝用メモリ100および102、1つのチェックノード計算器101、1つのバリエابلノード計算器103からなる。

【0041】

図9の復号装置では、枝用メモリ100または102からメッセージデータが一つずつ読み出され、そのメッセージデータを用いて、所望の枝に対応するメッセージデータが計算される。そして、その計算によって求められたメッセージデータが一つずつ後段の枝用メモリ102または100に格納されていく。繰り返し復号を行う際には、この1回復号を行う図9の復号装置を複数個縦列に接続するか、もしくは図9の復号装置を繰り返し用いることによって、繰り返し復号を実現する。なお、ここでは、例えば、図9の復号装置が複数個接続されているものとする。

【0042】

枝用メモリ100は、前段の復号装置(図示せず)のバリエابلノード計算器103から供給される出力メッセージD100を、後段のチェックノード計算器101が読み出す順番に格納していく。そして、枝用メモリ100は、チェックノード計算のフェーズでは、メッセージD100を、格納してある順番通りに、メッセージ出力D101として、チェックノード計算器101に供給する。チェックノード計算器101は、枝用メモリ100から供給されるメッセージD101を用いて、式(7)に従って演算を行い、その演算によって求められたメッセージD102を、後段の枝用メモリ102に供給する。

【0043】

図10は、チェックノード計算を一つずつ行う図9のチェックノード計算器1

01の構成例を示している。

【0044】

図10のチェックノード計算器101では、枝用メモリ100から供給される、検査行列の各列に対応するバリエブルノードからのメッセージ v_i を一つずつ読み込み、式(7)における $\phi(|v_i|)$ の演算をLUTによって行う。さらに、検査行列の1行に亘る各列に対応するバリエブルノードからのメッセージ v_i から求められた $\phi(|v_i|)$ が積算され、これにより、全ての枝からのメッセージ v_i から求められた $\phi(|v_i|)$ の積算値が求められる。その後、その積算値から、メッセージ u_j を求めたい枝から求められ、FIFO(FIFOメモリ)で遅延された $\phi(|v_i|)$ が減算され、これにより、メッセージ u_j を求めたい枝について、式(7)における $\sum \phi(|v_i|)$ が求められる。即ち、チェックノードへの枝すべてからのメッセージの和から、メッセージ u_j を求めたい枝からのメッセージを減算することで、メッセージ u_j を求めたい枝へのメッセージが求められる。さらに、LUTによって、式(7)における $\phi^{-1}(\sum \phi(|v_i|))$ の演算が行われる。同時に、メッセージ u_j の符号ビット、即ち、式(7)における $\Pi \text{sign}(v_i)$ も、EXOR回路を用いて同様に計算される。以上のようにして、式(7)の演算が行われ、メッセージ u_j が求められる。

【0045】

なお、図10では、各メッセージが符号ビットを合わせて合計6ビット(bit)に量子化されているものとして、チェックノード計算器101を表している。また、ここで処理の対象としている図8の検査行列の行の重み(row weight)の最大は9であるため、即ち、チェックノードに供給されるメッセージの最大数は9であるため、チェックノード計算器101は、9個のメッセージ($\phi(|v_i|)$)を遅延させるFIFO(First In First Out)を持っている。

【0046】

枝用メモリ102は、前段のチェックノード計算器101から供給される出力メッセージD102を、後段のバリエブルノード計算器103が読み出す順番に格納していく。そして、枝用メモリ102は、バリエブルノード計算のフェーズでは、メッセージ出力D102を、格納してある順番通りに、メッセージ出力D103として、バリエブルノード計算器103に供給する。

【0047】

バリエブルノード計算器103は、枝用メモリ102から供給されるメッセージD103と受信用メモリ104から供給される受信データD104を用いて式(1)に従って演算を行い、その演算の結果得られるメッセージD105を、図示せぬ後段の復号装置の枝用メモリ100に供給する。

【0048】

図11は、バリエブルノード計算を一つずつ行う図9のバリエブルノード計算器103の構成例を示している。

【0049】

図11のバリエブルノード計算器103では、枝用メモリ102から供給される、検査行列の各行に対応するチェックノードからのメッセージ u_j を一つずつ読み込み、検査行列の1列に亘る各行に対応するチェックノードからのメッセージを積算して、その積算値を求める。その後、その積算値から、メッセージ v_i を求めたい枝から供給され、FIFOで遅延されたメッセージが減算される。さらに、その結果得られる減算値から、受信値 u_{0i} を加算することで、式(1)の演算が行われ、これにより、メッセージ v_i が求められる。即ち、バリエブルノードへの枝すべてからのメッセージの和から、メッセージ v_i を求めたい枝からのメッセージを減算することで、メッセージ v_i を求めたい枝へのメッセージが求められる。

【0050】

図11においても、図10における場合と同様に、各メッセージが符号ビットを合わせて合計6ビットに量子化されているものとして、バリエブルノード計算器103を表している。また、ここで処理の対象としている図8の検査行列においては、列の重み(column weight)の最大値が5であるため、バリエブル計算器103は、5個のメッセージを遅延させるFIFOを持っており、列の重みが5未満の列のメッセージを計算するときには、FIFOにおける遅延量が、その列の重みの値に減らされる。

【0051】

図9の復号装置には、検査行列の重みにしたがって、制御信号が与えられる。そして、図9の復号装置によれば、枝用メモリ100および102、並びにチェ

ックノード計算器 101 およびバリエブルノード計算器 103 のFIFOの容量さえ足りれば、制御信号のみを変えることで様々な符号を復号することができる。

【0052】

なお、図示しないが、図9の復号装置において、復号の最終段においては、式(1)のバリエブルノード演算の代わりに、式(5)の演算が行われ、その演算結果が復号結果として出力される。

【0053】

図9の復号装置を繰り返し用いて、LDPC符号を復号する場合には、チェックノード演算とバリエブルノード演算とが交互に行われるため、269の枝を有する図8の検査行列を用いた1回の復号に、 $269 \times 2 = 538$ クロック(clock)を必要とする。従って、例えば、50回の繰り返し復号を行うためには、符号長である90個の符号情報を受信する間に、 $538 \times 50 = 26900$ クロック動作することが必要であり、受信周波数の約300 ($\div 26900/90$) 倍の高速動作が必要になる。受信周波数が数十MHzであるとする、GHz以上の速度での動作を要求されることになり実装は容易ではない。

【0054】

また、図9の復号装置を、例えば、50台接続して、LDPC符号を復号する場合には、1フレーム(frame)目がバリエブルノード演算を行っている間に、2フレーム目はチェックノード演算を行い、3フレーム目は前段のバリエブルノード演算を行う、というように、複数のバリエブルノード演算とチェックノード演算とを同時に行うことができる。この場合、90個の符号情報を受信する間に、269個の枝を計算すればよいので、復号装置は、受信周波数の約3 ($\div 269/90$) 倍の周波数で動作すればよいことになり、十分に実現可能である。しかしながら、この場合、回路規模が、単純には、図9の復号装置の50倍になる。

【0055】

次に、全ノードの演算を同時に行うことによって復号を行う場合(full parallel decoding)の復号装置の実装法について説明する。

【0056】

この実装法については、例えば、非特許文献1に記載されている。

【0057】

図12は、図8の検査行列で表現される符号（符号化率2/3、符号長90）を復号する復号装置の一例の構成を示している。

【0058】

図12の復号装置では、枝用メモリ202または206から、269個ある枝に対応するメッセージデータを全て同時に読み出し、そのメッセージデータを用いて、269個の枝に対応する新たなメッセージデータを演算する。さらに、その演算の結果求められた新たなメッセージデータが全て同時に後段の枝用メモリ206または202に格納されていく。そして、図12の復号装置を繰り返し用いることで繰り返し復号が実現される。

【0059】

図12において、復号装置は、1つの受信用メモリ205、2つの枝入れ替え装置200および203、2つの枝用メモリ202および206、30個のチェックノード計算器201₁乃至201₃₀、90個のバリエブルノード計算器204₁乃至204₉₀からなる。以下、各部について詳細に説明する。

【0060】

枝用メモリ206は、前段のバリエブルノード計算器204₁乃至204₉₀からの出力メッセージD206₁乃至D206₉₀を全て同時に格納し、次の時刻（次のクロックのタイミング）に、メッセージD206₁乃至D206₉₀を、メッセージD207₁乃至D207₉₀として読み出し、次段の枝入れ替え装置200に、メッセージD200（D200₁乃至D200₉₀）として供給する。枝入れ替え装置200は、枝用メモリ206から供給されたメッセージD200₁乃至D200₉₀の順番を、図8の検査行列に従って並び替え（入れ替え）、チェックノード計算器201₁乃至201₃₀に、メッセージD201₁乃至D201₃₀として供給する。

【0061】

チェックノード計算器201₁乃至201₃₀は、枝入れ替え装置200から供給されるメッセージD201₁乃至D201₃₀を用いて式（7）に従って演算を行い、その演算の結果得られるメッセージD202₁乃至D202₃₀を、枝用メモリ202に供給する。

【0062】

図13は、チェックノード演算を同時に行う図12のチェックノード計算器201_m (m=1, 2, ..., 30) の構成例を示している。

【0063】

図13のチェックノード計算器201_mでは、図10のチェックノード計算器101と同様にして、式(7)のチェックノード演算が行われるが、そのチェックノード演算が、すべての枝について同時に行われる。

【0064】

即ち、図13のチェックノード計算器201_mでは、枝入れ替え装置200から供給される図8の検査行列の各列に対応するバリエブルノードからのメッセージが全て同時に読み込まれ、式(7)における $\phi(|v_i|)$ の演算がLUTによって行われる。さらに、検査行列の1行に亘る各列に対応するバリエブルノードからのメッセージ v_i から求められた $\phi(|v_i|)$ が積算され、これにより、全ての枝からのメッセージ v_i から求められた $\phi(|v_i|)$ の積算値が求められる。その後、その積算値から、メッセージ u_j を求めたい枝から求められた $\phi(|v_i|)$ が減算され、これにより、メッセージ u_j を求めたい枝について、式(7)における $\Sigma \phi(|v_i|)$ が求められる。即ち、チェックノードへの枝すべてからのメッセージの和から、メッセージ u_j を求めたい枝からのメッセージを減算することで、メッセージ u_j を求めたい枝へのメッセージが求められる。さらに、LUTによって、式(7)における $\phi^{-1}(\Sigma \phi(|v_i|))$ の演算が行われる。同時に、メッセージ u_j の符号ビット、即ち、式(7)における $\Pi \text{sign}(v_i)$ も、EXOR回路を用いて同様に計算される。以上のようにして、式(7)の演算が行われ、メッセージ u_j が求められる。

【0065】

なお、図13では、各メッセージが符号ビットを合わせて合計6ビットに量子化されているものとして、チェックノード計算器201_mを表している。また、図13の回路は一つのチェックノードに相当する。ここで処理の対象としている図8の検査行列については、その行数である30行のチェックノードが存在するから、図12の復号装置は、図13に示したようなチェックノード計算器201_mを30個有している。

【0066】

ここで、図13のチェックノード計算器201_mでは、9個のメッセージを同時に計算することができる。そして、ここで処理の対象としている図8の検査行列の行の重みは、第1行が8で、第2乃至第30行が9であるため、即ち、チェックノードに供給されるメッセージの数が、8のケースが1つと、9のケースが29あるため、チェックノード計算器201₁は、図13の回路と同様の8つのメッセージを同時に計算することができる回路構成となっており、残りのチェックノード計算器201₂乃至201₃₀は、図13の回路と同一構成となっている。

【0067】

枝用メモリ202は、前段のチェックノード計算器201₁乃至201₃₀から供給される出力メッセージD202₁乃至D202₃₀を全て同時に格納し、次の時刻に、そのすべてのメッセージD202₁乃至D202₃₀を、出力メッセージD203₁乃至D203₃₀として、次段の枝入れ替え装置203に供給する。

【0068】

枝入れ替え装置203は、枝用メモリ202から供給されたメッセージD203₁乃至D203₃₀の順番を図8の検査行列に従って並び替え、バリアブルノード計算器204₁乃至204₉₀に、メッセージD204₁乃至D204₉₀として供給する。

【0069】

バリアブルノード計算器204₁乃至204₉₀は、枝入れ替え装置203から供給されるメッセージD204₁乃至D204₉₀と、受信用メモリ205から供給される受信データD205₁乃至D205₉₀を用いて式(1)に従って演算を行い、その演算の結果得られるメッセージD206₁乃至D206₉₀を、次段の枝用メモリ206に供給する。

【0070】

図14は、バリアブルノード演算を同時に行う図12のバリアブルノード計算器204_p ($p=1, 2, \dots, 90$)の構成例を示している。

【0071】

図14のバリアブルノード計算器204_pでは、図11のバリアブルノード計

算器 103 と同様にして、式 (7) のチェックノード演算が行われるが、そのチェックノード演算が、すべての枝について同時に行われる。

【0072】

即ち、図 14 のバリエブルノード計算器 204_p では、枝入れ替え装置 203 から供給される、検査行列の各行に対応するチェックノードからのメッセージ u_j が全て同時に読み込まれ、検査行列の 1 列に亘る各行に対応するチェックノードからのメッセージが積算されて、その積算値が求められる。その後、その積算値から、メッセージ v_i を求めたい枝から供給されたメッセージが減算され、その結果得られる減算値から、受信値 u_{0i} を加算することで、式 (1) の演算が行われ、これにより、メッセージ v_i が求められる。即ち、バリエブルノードへの枝すべてのからのメッセージの和から、メッセージ v_i を求めたい枝からのメッセージを減算することで、メッセージ v_i を求めたい枝へのメッセージが求められる。

【0073】

なお、図 14 では、各メッセージが符号ビットを合わせて合計 6 ビットに量子化されているものとして、バリエブルノード計算器 204_p を表している。また、図 14 の回路は一つのバリエブルノードに相当する。ここで処理の対象としている図 8 の検査行列については、その列数である 90 列のバリエブルノードが存在するから、図 12 の復号装置は、図 14 に示したような回路を 90 個有している。

【0074】

ここで、図 14 のバリエブルノード計算器 204_p では、5 個のメッセージを同時に計算することができる。そして、ここで処理の対象としている図 8 の検査行列は、重みが 5, 3, 2, 1 の列が、それぞれ、15 列、45 列、29 列、1 列あるので、バリエブルノード計算器 204₁ 乃至 204₉₀ のうちの 15 個は、図 14 の回路と同一構成となっており、残りの 45 個、29 個、1 個は、図 14 の回路と同様の 3, 2, 1 つのメッセージをそれぞれ同時に計算することができる回路構成となっている。

【0075】

なお、図示しないが、図 12 の復号装置においても、図 9 における場合と同様

に、復号の最終段においては、式(1)のバリエブルノード演算の代わりに、式(5)の演算が行われ、その演算結果が復号結果として出力される。

【0076】

図12の復号装置によれば、269個ある枝に対応するメッセージすべてを1クロックで同時に計算することができる。

【0077】

図12の復号装置を繰り返し用いて復号する場合には、チェックノード演算とバリエブルノード演算とを交互に行い、1回の復号を2クロックで行うことができる。従って、例えば、50回の復号を行うためには、90個の符号情報を受信する間に $2 \times 50 = 100$ クロック動作すれば良いことになり、ほぼ受信周波数と同一の動作周波数でよいことになる。一般的に、LDPC符号は、符号長が数千から数万と大きいことから、図12の復号装置を用いれば、復号回数を極めて多くすることができ、誤り訂正性能の向上が期待できる。

【0078】

しかしながら、図12の復号装置は、タナグラフのすべての枝に対応するメッセージの演算を、並列で行うため、回路規模が、符号長に比例して大きくなる。また、図12の復号装置を、ある符号長の、ある符号化率の、ある検査行列を持つLDPC符号の復号を行う装置として構成した場合、その復号装置において、他の符号長や、他の符号化率、他の検査行列を持つLDPC符号の復号を行うことは困難となる。即ち、図12の復号装置は、図9の復号装置のように、制御信号を変えただけでは、様々な符号を復号することはできず、符号依存性が高い。

【0079】

図9および図12の復号装置の他に、一つでも全てでもなく、4つずつのメッセージの計算を同時に行う実装法について、例えば、非特許文献2に述べられているが、この場合、メモリの異なるアドレスからの同時読み出し、もしくは同時書き込みを避けることが一般的には容易でなく、メモリアクセス制御が困難であるという問題がある。

【0080】

また、サムプロダクトアルゴリズムを近似して実装する方法なども提案されて

いるが、この方法では、性能の劣化を招いてしまう。

【0081】

【非特許文献1】

C. Howland and A. Blanksby, "Parallel Decoding Architectures for Low Density Parity Check Codes", Symposium on Circuits and Systems, 2001

【0082】

【非特許文献2】

E. Yeo, P. Pakzad, B. Nikolic and V. Anantharam, "VLSI Architectures for iterative Decoders in Magnetic Recording Channels", IEEE Transactions on Magnetics, Vol. 37, No. 2, March 2001

【0083】

【発明が解決しようとする課題】

サムプロダクトアルゴリズムをハードウェアに実装する場合には、上述したように、枝に対応するメッセージの演算（チェックノード演算とビットノード(bit node)計算）を、1つずつシリアル(serial)に行う方法、すべて並列（フルパラレル(full parallel))に行う方法、幾つかずつ並列（パラレル(parallel))に行う方法がある。

【0084】

しかしながら、枝に対応するメッセージの演算を1つずつ行う方法では、高い動作周波数が必要となる。そこで、スループット(throughput)を上げる方法として、装置を、パイプライン(pipeline)化する方法があるが、この場合、回路規模、特にメモリ（の容量）が大きくなってしまう。

【0085】

また、メッセージの演算を全て並列に行う方法では、ロジック(logic)の回路規模が大きくなるとともに、符号依存性が高くなる。

【0086】

さらに、メッセージの演算を、幾つかずつ並列に行う方法では、メモリアクセスの制御が難しくなる。

【0087】

本発明は、このような状況に鑑みてなされたものであり、ロジック、メモリ共に回路規模を抑制しつつ、動作周波数も十分実現可能な範囲に抑え、メモリアクセスの制御も容易に行うことができるようにするものである。

【0088】

【課題を解決するための手段】

本発明の復号装置は、 $P \times P$ の単位行列、その単位行列のコンポーネントである1のうちの1個以上が0になった行列である準単位行列、単位行列もしくは準単位行列をサイクリックシフトした行列であるシフト行列、単位行列、準単位行列、もしくはシフト行列のうちの複数の和である和行列、または $P \times P$ の0行列を構成行列として、LDPC符号の検査行列が、複数の構成行列の組合せで表される場合において、LDPC符号の復号のためのP個のチェックノードの演算を同時に行うチェックノード計算手段と、LDPC符号の復号のためのP個のバリエブルノードの演算を同時に行うバリエブルノード計算手段とを備えることを特徴とする。

【0089】

本発明においては、 $P \times P$ の単位行列、その単位行列のコンポーネントである1のうちの1個以上が0になった行列である準単位行列、単位行列もしくは準単位行列をサイクリックシフトした行列であるシフト行列、単位行列、準単位行列、もしくはシフト行列のうちの複数の和である和行列、または $P \times P$ の0行列を構成行列として、LDPC符号の検査行列が、複数の構成行列の組合せで表される場合において、LDPC符号の復号のためのP個のチェックノードの演算と、P個のバリエブルノードの演算とが同時に行われる。

【0090】

【発明の実施の形態】

以下に本発明の実施の形態を説明するが、請求項に記載の構成要件と、発明の実施の形態における具体例との対応関係を例示すると、次のようになる。この記載は、請求項に記載されている発明をサポートする具体例が、発明の実施の形態に記載されていることを確認するためのものである。従って、発明の実施の形態中には記載されているが、構成要件に対応するものとして、ここには記載されていない具体例があったとしても、そのことは、その具体例が、その構成要件に対

応するものではないことを意味するものではない。逆に、具体例が構成要件に対応するものとしてここに記載されていたとしても、そのことは、その具体例が、その構成要件以外の構成要件には対応しないものであることを意味するものでもない。

【0091】

さらに、この記載は、発明の実施の形態に記載されている具体例に対応する発明が、請求項に全て記載されていることを意味するものではない。換言すれば、この記載は、発明の実施の形態に記載されている具体例に対応する発明であって、この出願の請求項には記載されていない発明の存在、すなわち、将来、分割出願されたり、補正により追加される発明の存在を否定するものではない。

【0092】

請求項1に記載の復号装置は、

LDPC(Low Density Parity Check)符号の復号装置であって、

$P \times P$ の単位行列、その単位行列のコンポーネントである1のうちの1個以上が0になった行列である準単位行列、前記単位行列もしくは準単位行列をサイクリックシフトした行列であるシフト行列、前記単位行列、準単位行列、もしくはシフト行列のうちの複数の和である和行列、または $P \times P$ の0行列を構成行列として、前記LDPC符号の検査行列が、複数の前記構成行列の組合せで表される場合において、

前記LDPC符号の復号のためのP個のチェックノードの演算を同時に行うチェックノード計算手段（例えば、図16のチェックノード計算部302）と、

前記LDPC符号の復号のためのP個のバリエブルノードの演算を同時に行うバリエブルノード計算手段（例えば、図16のバリエブルノード計算部307）とを備えることを特徴とする。

【0093】

請求項2に記載の復号装置は、

前記チェックノード計算手段は、チェックノードの演算を行うP個のチェックノード計算器（例えば、図16のチェックノード計算器302₁乃至302₅）を有し、

前記バリエブルノード計算手段は、バリエブルノードの演算を行うP個のバリエブルノード計算器（例えば、図16のバリエブルノード計算器307₁乃至307₅）を有する

ことを特徴とする。

【0094】

請求項3に記載の復号装置は、

前記P個のチェックノードの演算、または前記P個のバリエブルノードの演算の結果得られるP個の枝に対応するメッセージデータを同時に読み書きするメッセージ記憶手段（例えば、図16の枝データ格納用メモリ300または304）をさらに備える

ことを特徴とする。

【0095】

請求項7に記載の復号装置は、

前記メッセージ記憶手段は、行数/P個のFIFO（例えば、図16のFIFO300₁乃至300₆）と、列数/P個のFIFO（例えば、図16のFIFO304₁乃至304₁₈）とで構成され、

前記行数/P個のFIFOと列数/P個のFIFOは、それぞれ、前記検査行列の行と列の重みに対応するワード数を有する

ことを特徴とする。

【0096】

請求項9に記載の復号装置は、

受信情報を格納するとともに、P個の前記受信情報を同時に読み出す受信情報記憶手段（例えば、図16の受信データ用メモリ306）をさらに備える

ことを特徴とする。

【0097】

請求項11に記載の復号装置は、

前記P個のチェックノードの演算、または前記P個のバリエブルノードの演算の結果得られるメッセージをサイクリックシフトするサイクリックシフト手段（例えば、図16のサイクリックシフト回路303または308）をさらに備える

ことを特徴とする。

【0098】

以下、本発明を適用した具体的な実施の形態について、図面を参照しながら詳細に説明する。

【0099】

図15は、 5×5 の行列の単位に間隔を空けた 30×90 の検査行列を示している。なお、図15の検査行列自体は、図8に示した検査行列と同一である。

【0100】

図15においては、検査行列は、 5×5 の単位行列、その単位行列の1のうち1個以上が0になった行列（以下、適宜、準単位行列という）、単位行列または準単位行列をサイクリックシフト(cyclic shift)した行列（以下、適宜、シフト行列という）、単位行列、準単位行列、またはシフト行列のうちの2以上の和（以下、適宜、和行列という）、 5×5 の0行列の組合わせで表わされている。なお、図15の検査行列で表現されるLDPC符号は、符号化率 $2/3$ 、符号長90である。

【0101】

図15の検査行列は、 5×5 の単位行列、準単位行列、シフト行列、和行列、0行列で構成されているといえることができる。そこで、検査行列を構成する、これらの 5×5 の行列を、以下、適宜、構成行列という。

【0102】

図16は、図15の検査行列で表現されるLDPC符号を復号する復号装置の一実施の形態の構成例を示している。

【0103】

図16では、復号装置は、6つのFIFO300₁乃至300₆からなる枝データ格納用メモリ300、FIFO300₁乃至300₆を選択するセクタ301、チェックノード計算器302、2つのサイクリックシフト回路303および308、18個のFIFO304₁乃至304₁₈からなる枝データ格納用メモリ304、FIFO304₁乃至304₁₈を選択するセクタ305、受信情報を格納する受信データ用メモリ306、バリアブルノード計算部307からなる。

【0104】

この復号装置の各部について詳細に説明する前に、まず、枝データ格納用メモリ300と304へのデータの格納方法について説明する。

【0105】

枝データ格納用メモリ300は、検査行列の行数30を構成行列の行数5で除算した数である6つのFIFO300₁乃至300₆から構成されている。FIFO300_y ($y=1, 2, \dots, 6$) は、構成行列の行数および列数である5つの枝に対応するメッセージを同時に読み出しもしくは書き込むことができるようになっており、その長さ(段数)は、検査行列の行方向の1の数(ハミング重み)の最大数である9になっている。

【0106】

FIFO300₁には、図15の検査行列の第1行目から第5行目までの1の位置に対応するデータが、各行共に横方向に詰めた形に(0を無視した形で)格納される。すなわち、第j行第i列を、(j, i)と表すこととすると、FIFO300₁の第1の要素(第1段)には、検査行列の(1, 1)から(5, 5)の5×5の単位行列の1の位置に対応するデータが格納される。第2の要素には、検査行列の(1, 21)から(5, 25)のシフト行列(5×5の単位行列を右方向に3つだけサイクリックシフトしたシフト行列)の1の位置に対応するデータが格納される。第3から第8の要素も同様に検査行列と対応づけてデータが格納される。そして、第9の要素には、検査行列の(1, 86)から(5, 90)のシフト行列(5×5の単位行列のうちの1行目の1を0に置き換えて1つだけ左にサイクリックシフトしたシフト行列)の1の位置に対応するデータが格納される。ここで、検査行列の(1, 86)から(5, 90)のシフト行列においては、1行目に1がないため、FIFO300₁の1行目のみ要素数は8、残りの行は要素数が9となる。

【0107】

FIFO300₂には、図15の検査行列の第6行目から第10行目までの1の位置に対応するデータが格納される。すなわち、FIFO300₂の第1の要素には、検査行列の(6, 1)から(10, 5)の和行列(5×5の単位行列を右に1つだけサイクリックシフトした第1のシフト行列と、右に2つだけサイクリックシフトした第

2のシフト行列の和である和行列)を構成する第1のシフト行列の1の位置に対応するデータが格納される。また、第2の要素には、検査行列の(6,1)から(10,5)の和行列を構成する第2のシフト行列の1の位置に対応するデータが格納される。

【0108】

即ち、重みが2以上の構成行列については、その構成行列を、重みが1である $P \times P$ の単位行列、そのコンポーネントである1のうち1個以上が0になった準単位行列、または単位行列もしくは準単位行列をサイクリックシフトしたシフト行列のうちの複数の和の形で表現したときの、その重みが1の単位行列、準単位行列、またはシフト行列の1の位置に対応するデータ(単位行列、準単位行列、またはシフト行列に属する枝に対応するメッセージ)は、同一アドレス(FIFO300₁乃至300₆のうちの同一のFIFO)に格納される

【0109】

以下、第3から第9の要素についても、検査行列に対応づけてデータが格納される。FIFO300₂は全行共に要素数は9となる。

【0110】

FIFO300₃乃至300₆も同様に検査行列に対応づけてデータを格納し、各FIFO300₃乃至300₆それぞれの長さは9である。

【0111】

枝データ格納用メモリ304は、検査行列の列数90を、構成行列の列数である5で割った18個のFIFO304₁乃至304₁₈から構成されている。FIFO304_x($x=1, 2, \dots, 18$)は、構成行列の行数および列数である5つの枝に対応するメッセージを同時に読み出しもしくは書き込むことができるようになっていてる。

【0112】

FIFO304₁には、図15の検査行列の第1列目から第5列目までの1の位置に対応するデータが、各列共に縦方向に詰めた形に(0を無視した形で)格納される。すなわち、FIFO304₁の第1の要素(第1段)には、検査行列の(1,1)から(5,5)の 5×5 の単位行列の1の位置に対応するデータが格納される。第2の

要素には、検査行列の(6,1)から(10,5)の和行列(5×5の単位行列を右に1つだけサイクリックシフトした第1のシフト行列と、右に2つだけサイクリックシフトした第2のシフト行列との和である和行列)を構成する第1のシフト行列の1の位置に対応するデータが格納される。また、第3の要素には、検査行列の(6,1)から(10,5)の和行列を構成する第2のシフト行列の1の位置に対応するデータが格納される。

【0113】

即ち、重みが2以上の構成行列については、その構成行列を、重みが1であるP×Pの単位行列、そのコンポーネントである1のうち1個以上が0になった準単位行列、または単位行列もしくは準単位行列をサイクリックシフトしたシフト行列のうちの複数の和の形で表現したときの、その重みが1の単位行列、準単位行列、またはシフト行列の1の位置に対応するデータ(単位行列、準単位行列、またはシフト行列に属する枝に対応するメッセージ)は、同一アドレス(FIFO304₁乃至304₁₈のうちの同一のFIFO)に格納される。

【0114】

以下、第4および第5の要素についても、検査行列に対応づけて、データが格納される。このFIFO304₁の要素数(段数)は、検査行列の第1列から第5列における行方向の1の数(ハミング重み)の最大数である5になっている。

【0115】

FIFO304₂と304₃も同様に検査行列に対応づけてデータを格納し、それぞれの長さ(段数)は、5である。FIFO304₄乃至304₁₂も同様に検査行列に対応づけてデータを格納し、それぞれの長さは3である。FIFO304₁₃乃至304₁₈も同様に検査行列に対応づけてデータを格納し、それぞれの長さは2である。但し、FIFO304₁₈の第1の要素は、検査行列の(1,86)から(5,90)に相当し、第5列目(検査行列の(1,90)から(5,90))に1がないため、データは格納されない。

【0116】

以下、図16の復号装置の各部の動作について詳細に説明する。

【0117】

枝データ格納用メモリ300は、6つのFIFO300₁乃至300₆からなり、前段のサイクリックシフト回路308から供給される5つのメッセージデータD311が、検査行列どの行に属するかの情報（Matrixデータ）D312に従って、データを格納するFIFOを、FIFO300₁乃至300₆の中から選び、選んだFIFOに5つのメッセージデータD311をまとめて順番に格納していく。また、枝データ格納用メモリ300は、データを読み出す際には、FIFO300₁から5つのメッセージデータD300₁を順番に読み出し、次段のセクタ301に供給する。枝データ格納用メモリ300は、FIFO300₁からのメッセージデータの読み出しの終了後、FIFO300₂乃至300₆からも、順番に、メッセージデータを読み出し、セクタ301に供給する。

【0118】

セクタ301は、セレクト信号D301に従って、FIFO300₁乃至300₆のうちの、現在データが読み出されているFIFOからの5つのメッセージデータを選択し、メッセージデータD302として、チェックノード計算部302に供給する。

【0119】

チェックノード計算部302は、5つのチェックノード計算器302₁乃至302₅からなり、セクタ301を通して供給されるメッセージD302(D302₁乃至D302₅)を用いて、式(7)に従って演算を行い、その演算の結果得られる5つのメッセージD303(D303₁乃至D303₅)をサイクリックシフト回路303に供給する。

【0120】

ここで、チェックノード計算器302₁乃至302₅それぞれは、図10に示したチェックノード計算器101と同様に構成される。

【0121】

サイクリックシフト回路303は、チェックノード計算部302で計算された5つのメッセージD303₁乃至D303₅を、対応する枝が検査行列において元となる単位行列を幾つサイクリックシフトしたものであるかの情報（Matrixデータ）D305を元にサイクリックシフトし、その結果をメッセージD304として、枝データ格納用メモリ304に供給する。

【0122】

枝データ格納用メモリ 304 は、18 個の FIFO 304₁ 乃至 304₁₈ からなり、前段のサイクリックシフト回路 303 から供給される 5 つのメッセージデータ D304 が検査行列のどの行に属するかの情報 D305 に従って、データを格納する FIFO を、FIFO 304₁ 乃至 304₁₈ の中から選び、選んだ FIFO に 5 つのメッセージデータ D304 をまとめて順番に格納していく。また、枝データ格納用メモリ 304 は、データを読み出す際には、FIFO 304₁ から 5 つのメッセージ D306₁ を順番に読み出し、次段のセクタ 305 に供給する。枝データ格納用メモリ 304 は、FIFO 304₁ からのデータの読み出しの終了後、FIFO 304₂ 乃至 304₁₈ からも、順番に、メッセージデータを読み出し、セクタ 305 に供給する。

【0123】

セクタ 305 は、セレクト信号 D307 に従って、FIFO 304₁ 乃至 304₁₈ のうちの、現在データが読み出されている FIFO からの 5 つのメッセージデータを選択し、メッセージデータ D308 として、バリアブルノード計算部 307 に供給する。

【0124】

一方、受信データ用メモリ 306 は、通信路を通して受信した受信信号から、受信 LLR (対数尤度比) を計算しており、その計算した受信 LLR を 5 つまとめてデータ D309 としてバリアブルノード計算部 307 に供給する。

【0125】

バリアブルノード計算部 307 は、5 つのバリアブルノード計算器 307₁ 乃至 307₅ からなり、セクタ 305 を通して供給されるメッセージ D308 (D308₁ 乃至 D308₅) と、受信データ用メモリ 306 から供給される 5 つの受信 LLR D309 を用いて式 (1) に従って演算を行い、その演算の結果得られるメッセージ D310 (D310₁ 乃至 D310₅) を、サイクリックシフト回路 308 に供給する。

【0126】

ここで、バリアブルノード計算器 307₁ 乃至 307₅ それぞれは、図 11 のバリアブルノード計算器 103 と同様に構成される。

【0127】

サイクリックシフト回路 308 は、バリアブルノード計算部 307 で計算され

たメッセージD310₁乃至D310₅を、対応する枝が検査行列において元となる単位行列を幾つサイクリックシフトしたものであるかの情報を元にサイクリックシフトし、その結果をメッセージD311として、枝データ格納用メモリ300に供給する。

【0128】

以上の動作を1巡することで、LDPC符号の1回の復号を行うことができる。図16の復号装置は、所定の回数だけLDPC符号を復号した後、図示しないが、式(5)に従って最終的な復号結果を求めて出力する。

【0129】

なお、枝データ(枝に対応するメッセージ)が欠けている箇所に関しては、メモリ格納時(枝データ格納用メモリ300と304へのデータ格納時)には、何のメッセージも格納せず、また、ノード演算時(チェックノード計算部302でのチェックノード演算時とバリアブルノード計算部307でのバリアブルノード演算時)にも何の演算も行わない。

【0130】

また、サイクリックシフト回路303および308には、バレルシフタを用いると回路規模を小さくしながら所望の操作を実現できる。

【0131】

上記説明には、枝データ格納にFIFOを用いたが(枝データ格納メモリ300と304をFIFOで構成するようにしたが)、FIFOの代わりにRAMを用いても構わない。その場合、RAMには、P個の枝情報(枝に対応するメッセージ)を同時に読み出すことの出来るビット幅と、枝総数/Pのワード(word)数が必要となる。さらに、RAMへの書き込みは、検査行列の情報から、書き込もうとしているデータが次に読み出される際に何番目に読み出されるかを求め、その位置に書き込む。また、RAMからの読み出しの際には、アドレスの先頭から順次データを読み出す。FIFOの代わりにRAMを用いると、セクタ301および305は不要になる。

【0132】

なお、FIFOやRAMの物理的なビット幅が足りない場合には、複数のRAMを用いて同じ制御信号を与えることで、論理的に1つのRAMとみなすことができる。

【0133】

また、上述の場合には、説明を簡単にするために、Pが5の場合、即ち、検査行列を構成する構成行列の行数および列数が5の場合を例に挙げたが、構成行列の行数および列数Pは必ずしも5である必要はなく、検査行列によって異なる値を取ることもあり得る。例えば、Pは360や392であってもよい。

【0134】

また、本実施の形態では、符号長90、符号化率2/3のLDPC符号を用いたが、LDP C符号の符号長や符号化率は、幾つであっても構わない。例えば、構成行列の行数および列数Pが5の場合、枝総数が5以下であれば、どんな符号長、符号化率のLDPC符号でも、制御信号を代えるだけで、図16の復号装置を用いて復号可能である。

【0135】

さらに、構成行列の行数および列数Pが所定の値で、枝の総数がある値以下、という条件を満たすあるLDPC符号の復号装置は、その条件を満たす、任意の符号長で、任意の符号化率のLDPC符号を復号することができる。

【0136】

検査行列が、構成行列の行数および列数Pの倍数でない場合は、検査行列の端数の外側にすべて0 (all 0) の成分を付けてPの倍数とみなして適用できることがある。

【0137】

次に、上述した一連の処理は、ハードウェアにより行うこともできるし、ソフトウェアにより行うこともできる。一連の処理をソフトウェアによって行う場合には、そのソフトウェアを構成するプログラムが、汎用のコンピュータ等にインストールされる。

【0138】

そこで、図17は、上述した一連の処理を実行するプログラムがインストールされるコンピュータの一実施の形態の構成例を示している。

【0139】

プログラムは、コンピュータに内蔵されている記録媒体としてのハードディスク

ク405やROM403に予め記録しておくことができる。

【0140】

あるいはまた、プログラムは、フレキシブルディスク、CD-ROM(Compact Disc Read Only Memory)、MO(Magneto Optical)ディスク、DVD(Digital Versatile Disc)、磁気ディスク、半導体メモリなどのリムーバブル記録媒体411に、一時的あるいは永続的に格納(記録)しておくことができる。このようなリムーバブル記録媒体411は、いわゆるパッケージソフトウェアとして提供することができる。

【0141】

なお、プログラムは、上述したようなリムーバブル記録媒体411からコンピュータにインストールする他、ダウンロードサイトから、デジタル衛星放送用の人工衛星を介して、コンピュータに無線で転送したり、LAN(Local Area Network)、インターネットといったネットワークを介して、コンピュータに有線で転送し、コンピュータでは、そのようにして転送されてくるプログラムを、通信部408で受信し、内蔵するハードディスク405にインストールすることができる。

【0142】

コンピュータは、CPU(Central Processing Unit)402を内蔵している。CPU402には、バス401を介して、入出力インタフェース410が接続されており、CPU402は、入出力インタフェース410を介して、ユーザによって、キーボードや、マウス、マイク等で構成される入力部407が操作等されることにより指令が入力されると、それにしたがって、ROM(Read Only Memory)403に格納されているプログラムを実行する。あるいは、また、CPU402は、ハードディスク405に格納されているプログラム、衛星若しくはネットワークから転送され、通信部408で受信されてハードディスク405にインストールされたプログラム、またはドライブ409に装着されたリムーバブル記録媒体411から読み出されてハードディスク405にインストールされたプログラムを、RAM(Random Access Memory)404にロードして実行する。これにより、CPU402は、上述したフローチャートにしたがった処理、あるいは上述したブロック図の構

成により行われる処理を行う。そして、CPU 402は、その処理結果を、必要に応じて、例えば、入出力インタフェース 410を介して、LCD(Liquid Crystal Display)やスピーカ等で構成される出力部 406から出力、あるいは、通信部 408から送信、さらには、ハードディスク 405に記録等させる。

【0143】

ここで、本明細書において、コンピュータに各種の処理を行わせるためのプログラムを記述する処理ステップは、必ずしもフローチャートとして記載された順序に沿って時系列に処理する必要はなく、並列的あるいは個別に実行される処理(例えば、並列処理あるいはオブジェクトによる処理)も含むものである。

【0144】

また、プログラムは、1のコンピュータにより処理されるものであっても良いし、複数のコンピュータによって分散処理されるものであっても良い。さらに、プログラムは、遠方のコンピュータに転送されて実行されるものであっても良い。

【0145】

以上のように、 $P \times P$ の単位行列、そのコンポーネントの1のうち1個以上が0になった準単位行列、単位行列もしくは準単位行列をサイクリックシフトしたシフト行列、単位行列、準単位行列、もしくはシフト行列の複数の和である和行列、 $P \times P$ の0行列の組合せで表わすことができる検査行列を持つLDPC符号の復号を、チェックノードとバリエブルノードの演算をP個同時に行うアーキテクチャ(architecture)を採用することにより、ノード演算を、P個同時に行うことで動作周波数を実現可能な範囲に抑えることができ、多数の繰り返し復号を行うことを可能にしつつ、メモリ(FIFOやRAM)への書き込みと読み出し時に、異なるアドレスへの同時アクセスが起きることを防止することができる。

【0146】

さらに、図16の復号装置を繰り返し用いて、図15の検査行列で表わされるLDPC符号を復号する場合には、269個の枝をチェックノード、バリエブルノード毎に5個ずつ演算することが可能であることから、1回の復号に、 $269/5 \times 2 = 108$ クロック動作すればよいことになる。50回の復号には、90個の符号情

報を受信する間に、 $108 \times 50 = 5400$ クロック動作すればよいことになり、受信周波数の約60倍の動作周波数でよいことになる。従って、図16の復号装置によれば、各ノード演算を一つずつ行う図9の復号装置に比べて、1/5の動作周波数で済むことになる。また、回路規模の面から見ても、メモリの大きさは同じであるため、論理回路が多少大きくなっても全体への影響は小さいと言える。

【0147】

一般的に、LDPC符号は符号長が数千から数万と大きいため、Pの値も数百の大きさを持つものが使われる。その場合には、更に本発明に係る復号装置を用いる効果は大きくなる。

【0148】

また、本発明に係る復号装置は、サンプロダクトアルゴリズムを忠実に実装するものであるため、メッセージの量子化以外の復号損失が起きることはない。

【0149】

以上の観点から、本発明に係る復号装置を用いることで、高性能な復号が可能になる。

【0150】

【発明の効果】

以上の如く、本発明によれば、回路規模を抑制しつつ、動作周波数も十分実現可能な範囲に抑え、メモリアクセスの制御も容易に行うことが可能となる。

【図面の簡単な説明】

【図1】

LDPC符号の検査行列Hを説明する図である。

【図2】

LDPC符号の復号手順を説明するフローチャートである。

【図3】

メッセージの流れを説明する図である。

【図4】

LDPC符号の検査行列の例を示す図である。

【図5】

検査行列のタナーグラフを示す図である。

【図 6】

バリエブルノードを示す図である。

【図 7】

チェックノードを示す図である。

【図 8】

LDPC符号の検査行列の例を示す図である。

【図 9】

ノード演算を一つずつ行うLDPC符号の復号装置の構成例を示すブロック図である。

【図 10】

メッセージを一つずつ計算するチェックノード計算器の構成例を示すブロック図である。

【図 11】

メッセージを一つずつ計算するバリエブルノード計算器の構成例を示すブロック図である。

【図 12】

ノード演算を全て同時に行うLDPC符号の復号装置の構成例を示すブロック図である。

【図 13】

メッセージを同時に計算するチェックノード計算器の構成例を示すブロック図である。

【図 14】

メッセージを同時に計算するバリエブルノード計算器の構成例を示すブロック図である。

【図 15】

5×5単位に分割した検査行列を示す図である。

【図 16】

本発明を適用した復号装置の一実施の形態の構成例を示すブロック図である。

【図 17】

本発明を適用したコンピュータの一実施の形態の構成例を示すブロック図である。

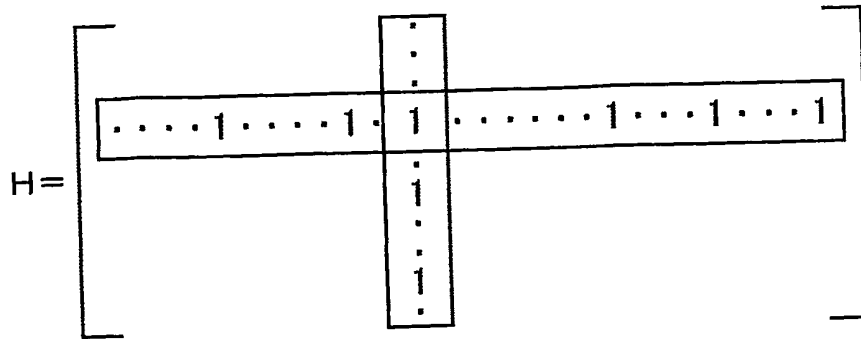
【符号の説明】

300 枝データ格納用メモリ, 301 セレクタ, 302 チェックノード計算部, 303 サイクリックシフト回路, 304 枝データ格納用メモリ, 305 セレクタ, 306 受信データ用メモリ, 307 バリアブルノード計算部, 308 サイクリックシフト回路, 401 バス, 402 CPU, 403 ROM, 404 RAM, 405 ハードディスク, 406 出力部, 407 入力部, 408 通信部, 409 ドライブ, 410 入出力インタフェース, 411 リムーバブル記録媒体

【書類名】 図面

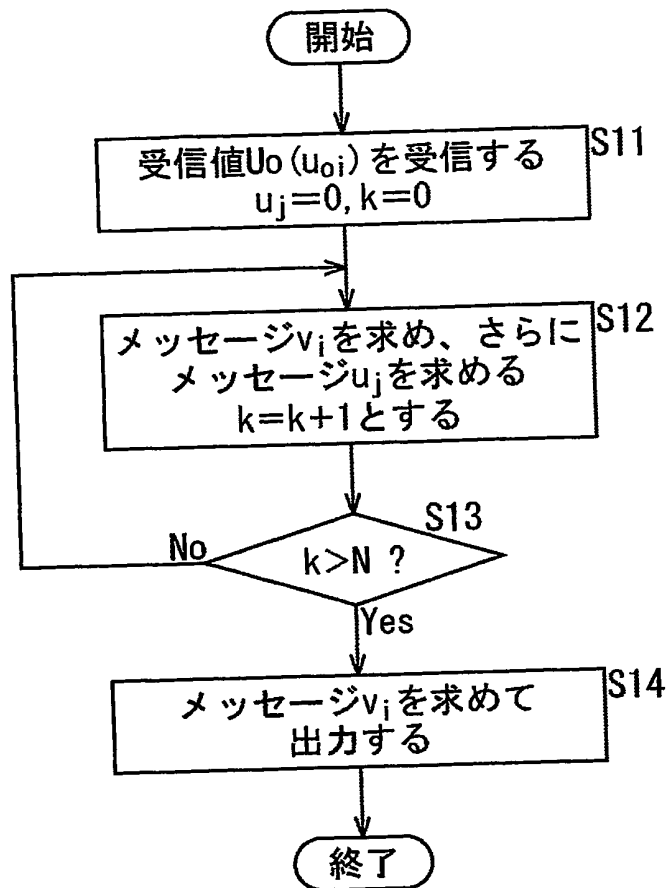
【図 1】

図1



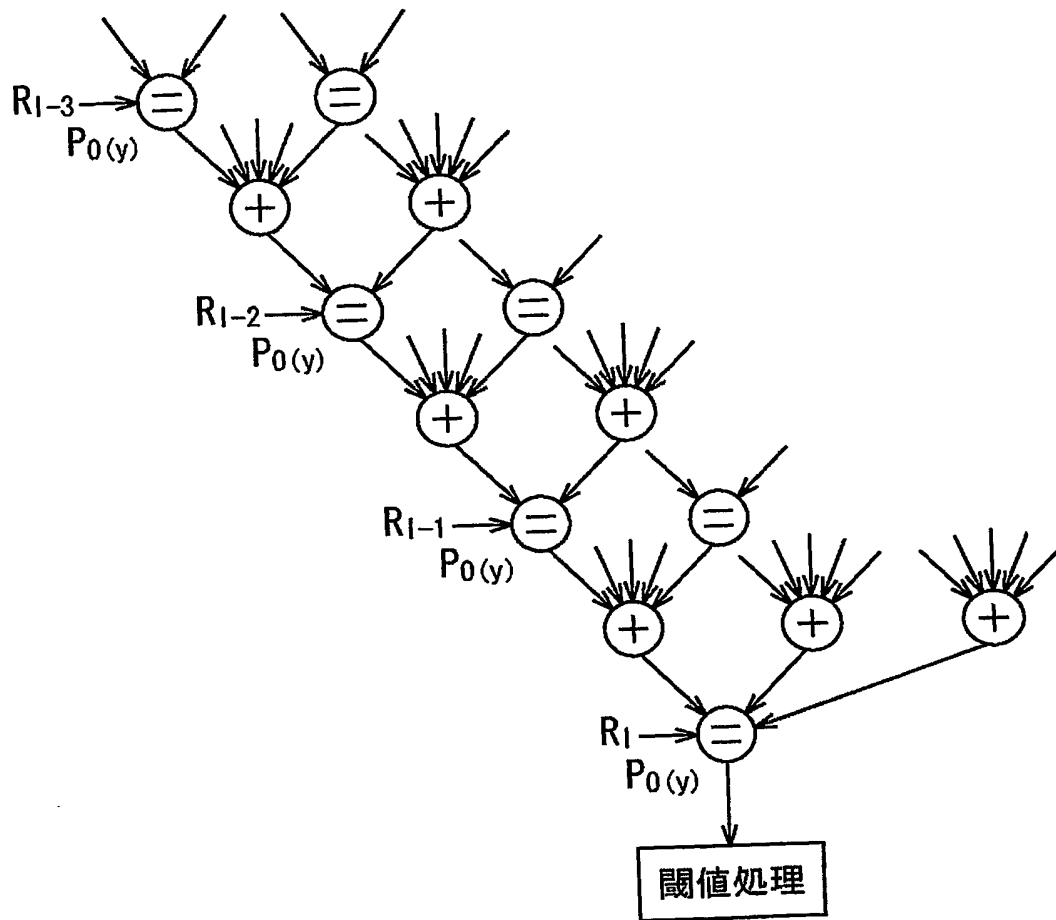
【図2】

図2



【図 3】

図3



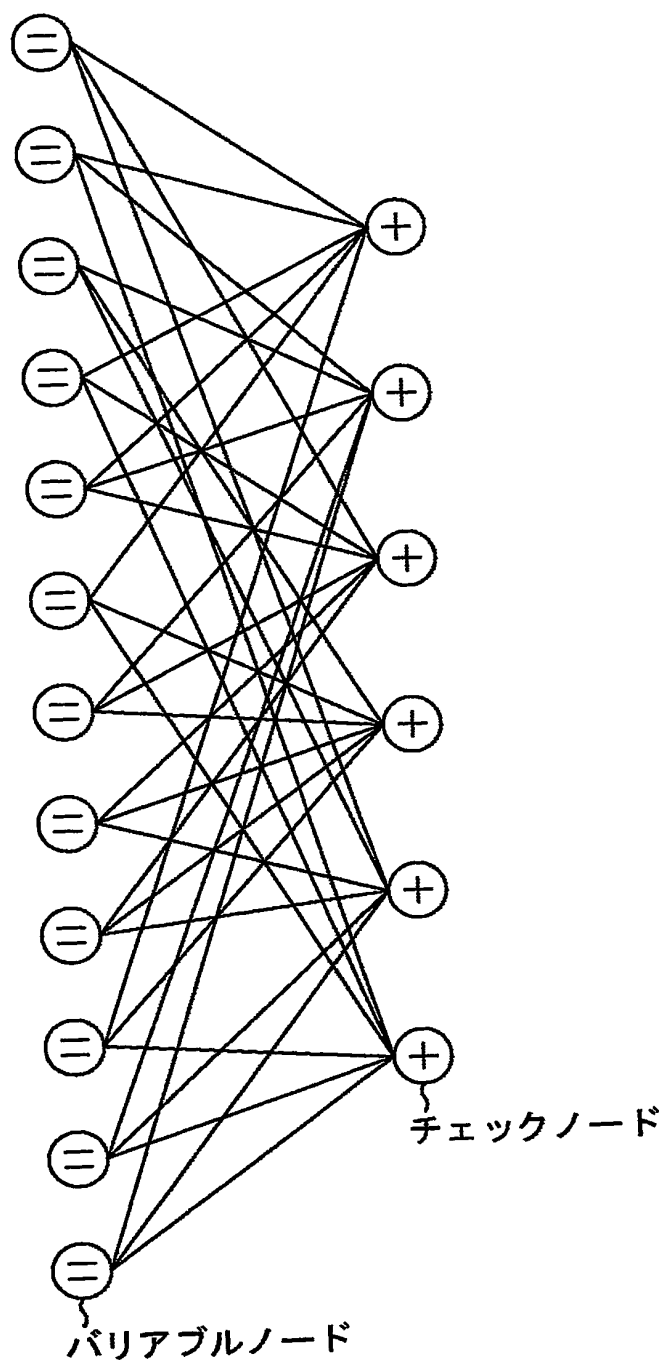
【図4】

図4

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \end{bmatrix}$$

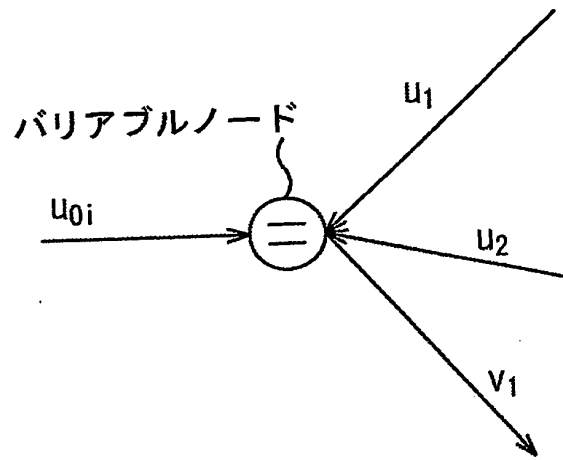
【図5】

図5



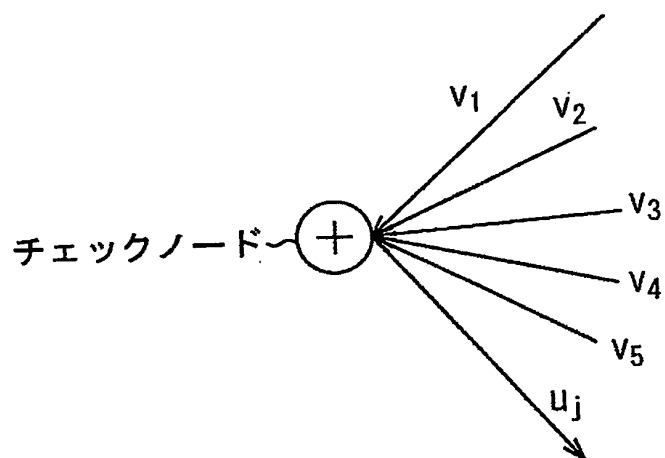
【図 6】

図6



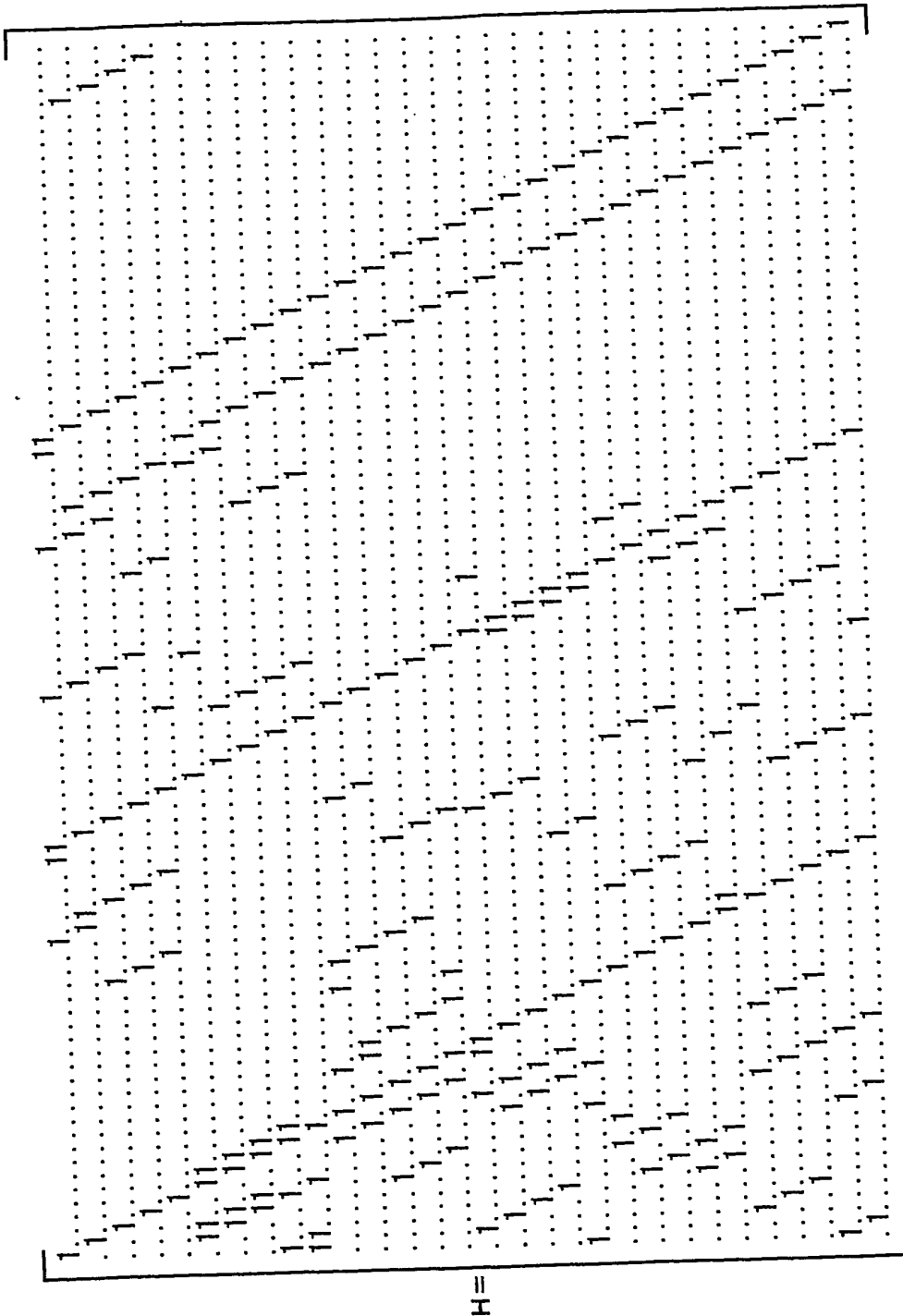
【図 7】

図 7



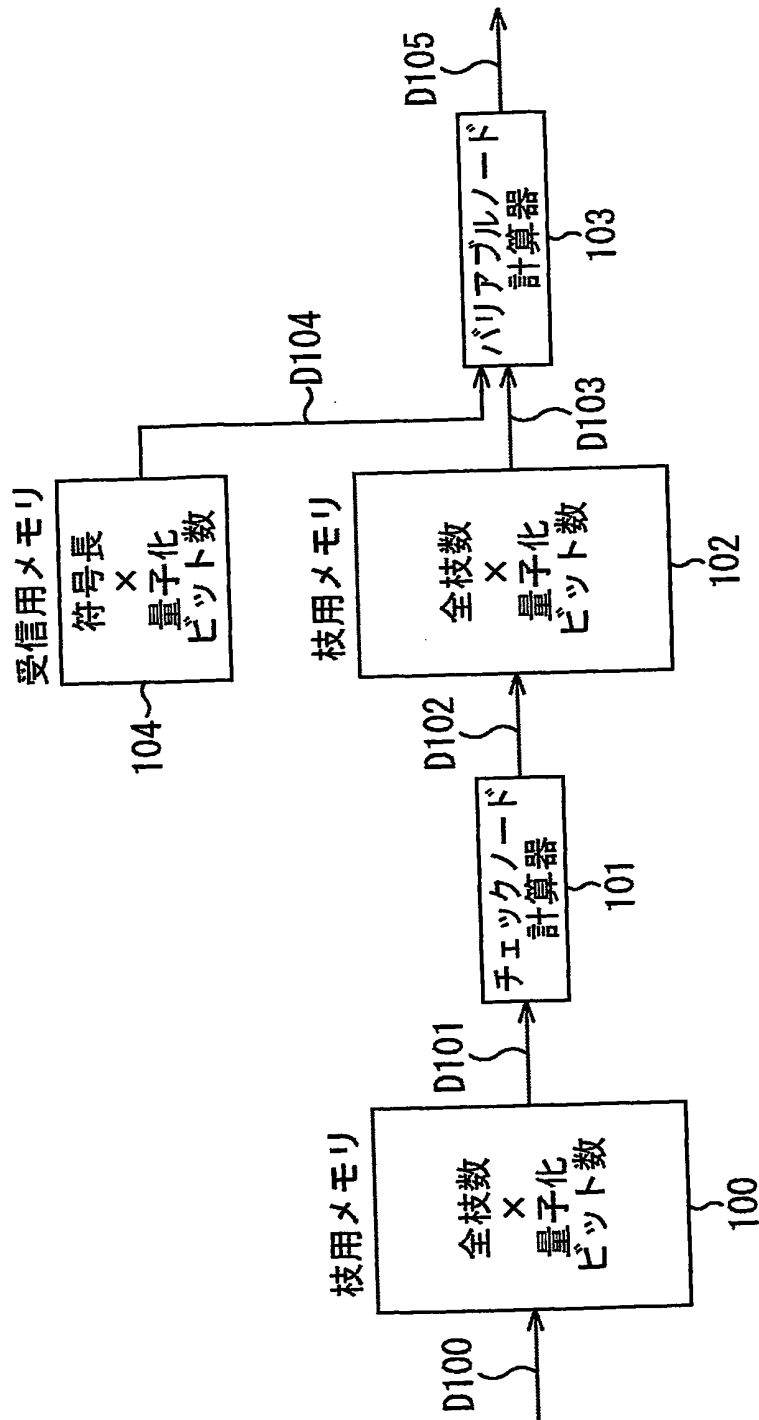
【図8】

図8



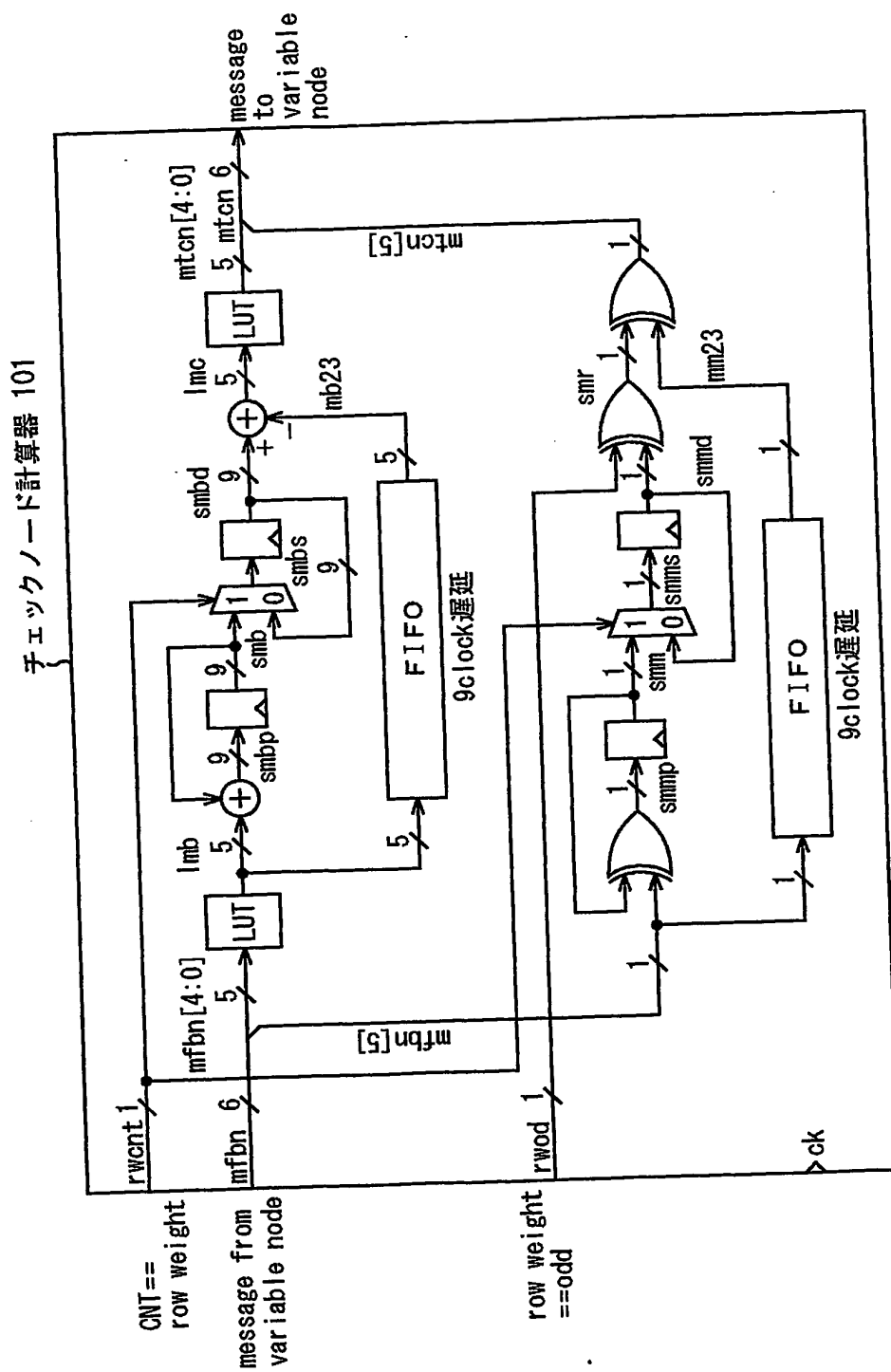
【図 9】

図9



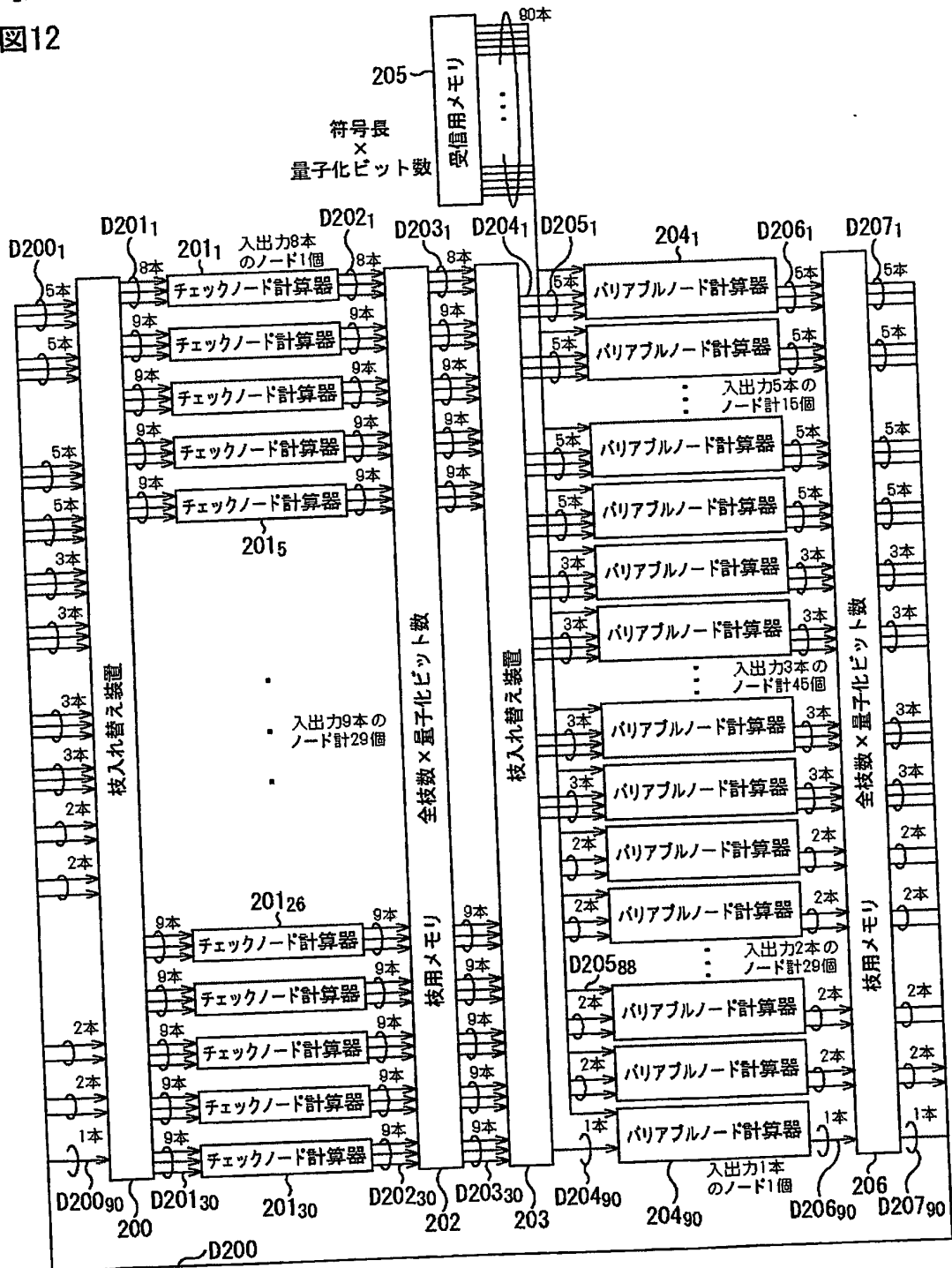
【図 10】

图10



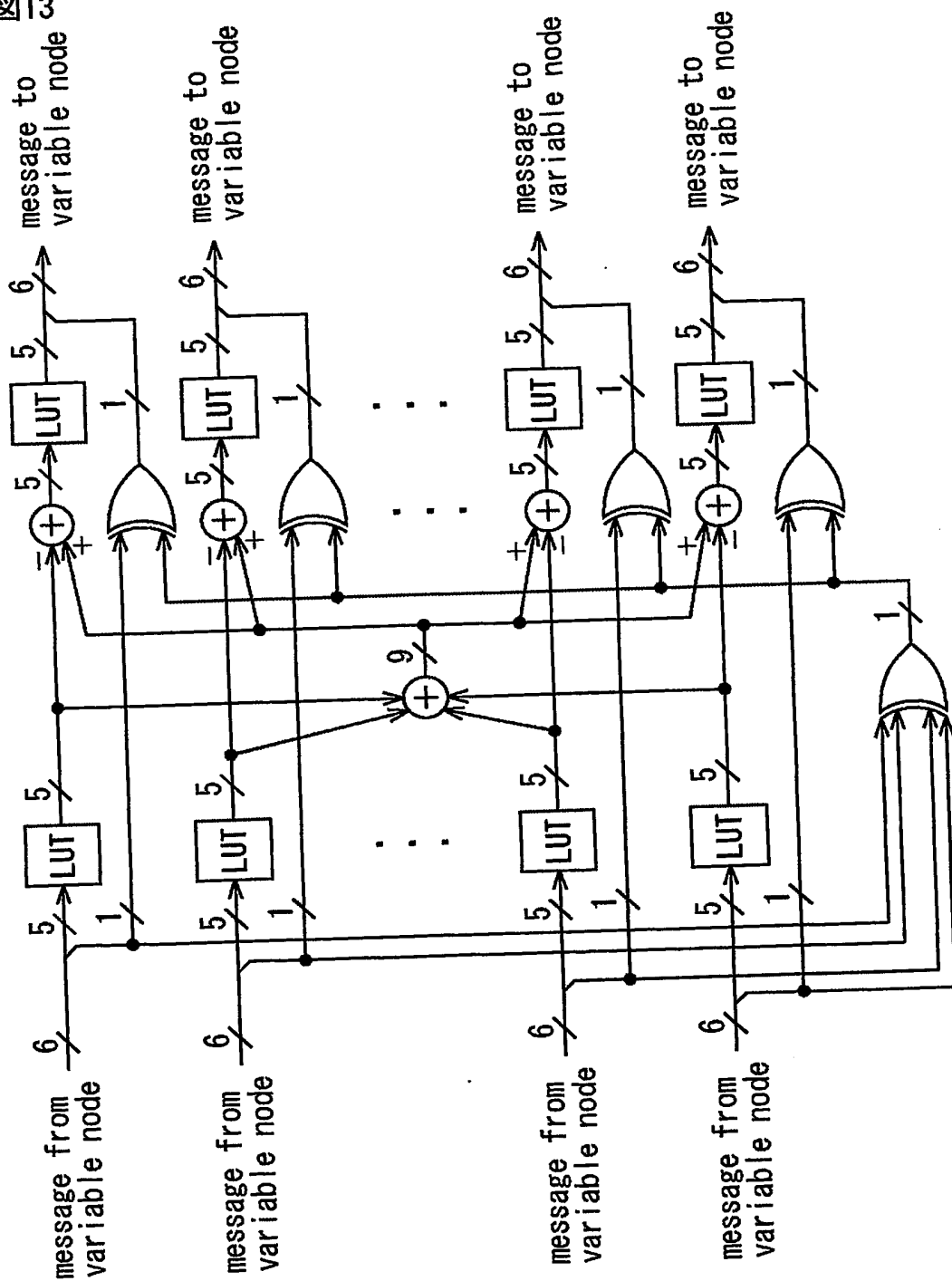
【図12】

図12



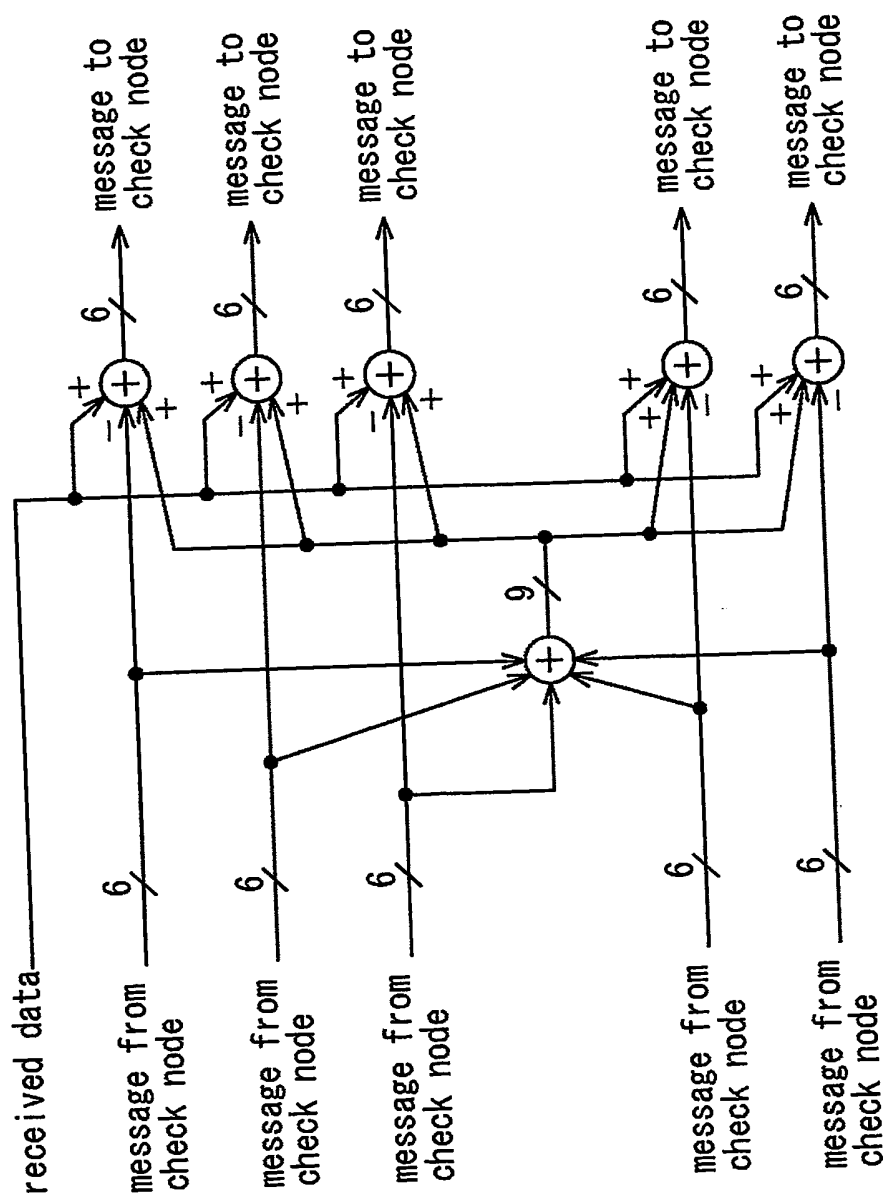
【図 13】

図 13

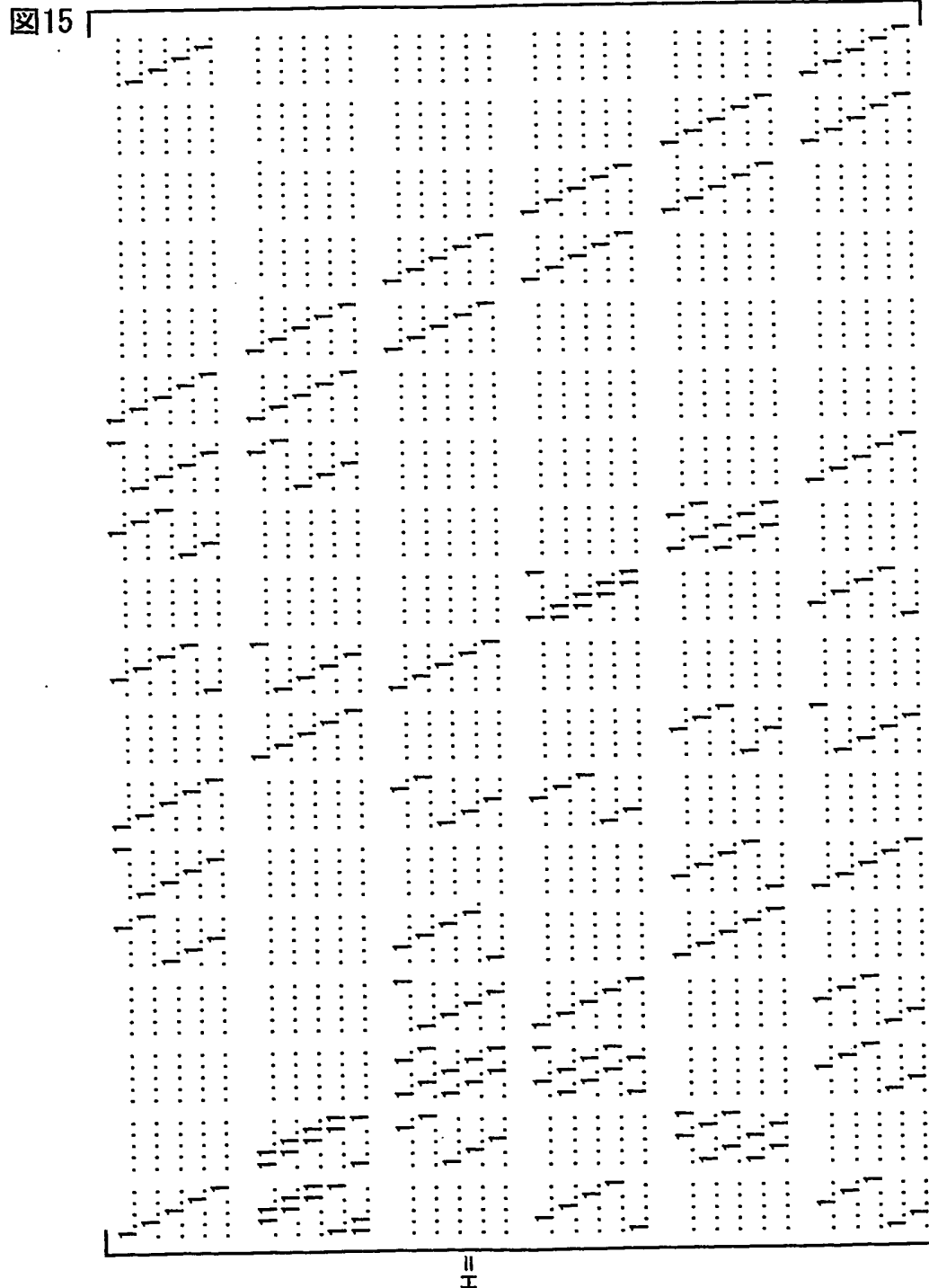


【図 14】

図14

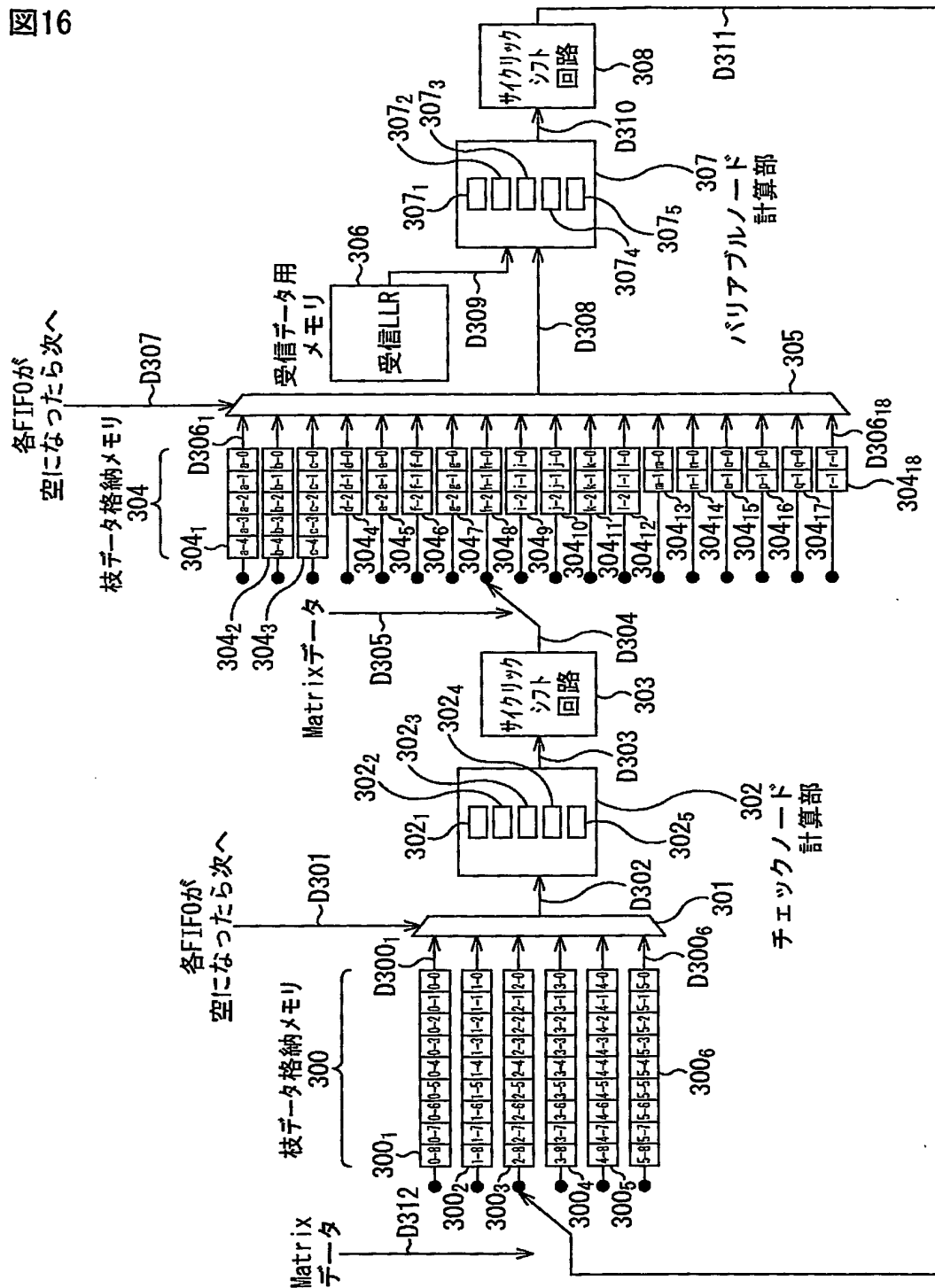


【図15】



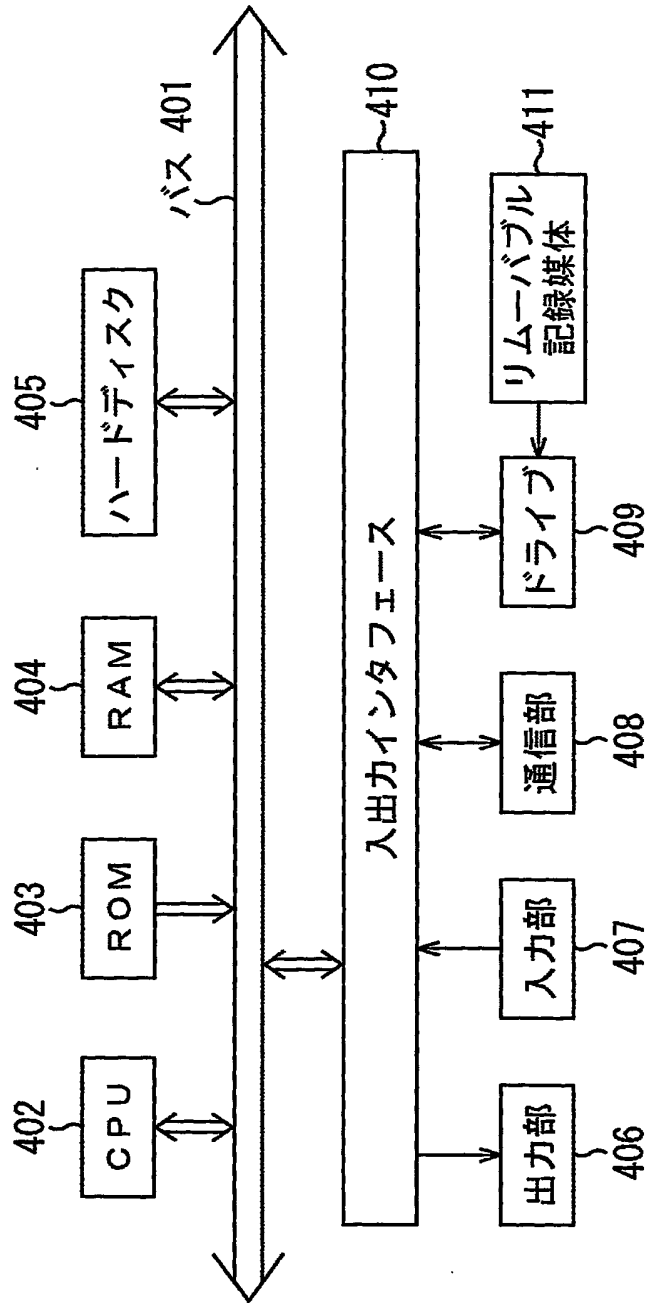
【図 16】

図16



【図 17】

図17



コンピュータ

【書類名】 要約書

【要約】

【課題】 回路規模を抑制しつつ、動作周波数も十分実現可能な範囲に抑え、メモリアクセスの制御も容易に行うことができるLDPC符号の復号を実現する。

【解決手段】 LDPC符号の検査行列は、 $P \times P$ の単位行列、その単位行列の1のうちの1個から数個が0になった行列、それらのサイクリックシフト、それらの複数の和、 $P \times P$ の0行列の組合せで構成される。チェックノード計算部302は、チェックノードの演算を、 P 個同時に行い、バリアブルノード計算部304は、バリアブルノードの演算を、 P 個同時に行う。

【選択図】 図16

特願 2 0 0 3 - 1 3 3 9 4 1

出 願 人 履 歴 情 報

識別番号

[0 0 0 0 0 2 1 8 5]

1. 変更年月日

1 9 9 0 年 8 月 3 0 日

[変更理由]

新規登録

住 所

東京都品川区北品川 6 丁目 7 番 3 5 号

氏 名

ソニー株式会社

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.